VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Rozšíření aplikace atop o komunikaci klient-server a uložení statistik

Improving Atop Application with Client-Server Communication and Statistics Storing

2013 Michal Reš

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Michal Reš**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Rozšíření aplikace atop o komunikaci klient-server a uložení statistik

Improving Atop Application with Client-Server Communication and Statistics Storing

Zásady pro vypracování:

Cílem práce je rozšíření monitoru výkonu atop o komunikaci klient-server a uložení statistik do zvoleného databázového systému. Klient bude komprimovat a zapisovat data do síťového rozhraní, server bude data číst a ukládat do databáze. Součástí aplikace bude i vytváření analýz z uložených statistik.

1. Seznamte se s aplikací atop a metodami komprimace vhodnými pro výstupní data monitoru výkonu atop.
2. Navrhněte schéma databáze pro tato výstupní data a proveďte fyzický návrh databáze pro PostgreSQL pro uvažovanou charakteristiku zátěže.
3. Navrhněte a naimplementujte klientské a serverové rozšíření atop obohacené o komprimovanou komunikaci a ukládání dat do databáze.
4. Navrhněte několik analýz, které je možné nad uloženými daty provést. Výsledky analýz generujte do vhodného reportovacího formátu.
5. Vyhodnoťte výsledky vytvořené aplikace.

Seznam doporučené odborné literatury:

Dokumentace SW atop, dokumentace PostgreSQL (dostupné online), dále dle pokynů vedoucího bakalářské práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Nikola Ciprich**

Datum zadání: 16.11.2012

Datum odevzdání: 07.05.2013

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne: *1. 5. 2013*

.....
podpis studenta

Rád bych poděkoval svému vedoucímu Ing. Nikolovi Ciprichovi, za ochotný přístup a odborné rady.

Abstrakt

Cílem této bakalářské práce je rozšíření aplikace Atop o komunikaci server-klient, kdy stávající aplikace bude doplněna o zápis na socket ta bude použita na straně klienta. Přijímací strana dále jen server, tato data přijme na socketu, nebo jako generovaný soubor Atopem, dekomprimuje je a uloží do databáze.

První část práce se zabývá popisem původní aplikace Atop, její funkčností a výstupem. V druhé kapitole je popsána analýza původní aplikace, použité technologie pro úpravu stávající aplikace a vývoj nové aplikace serveru. Dále je zde popsána implementace jednotlivých částí. V třetí kapitole je popsána datová vrstva aplikace, analýza, návrh a implementace databáze. V poslední kapitole je uveden proces testování celé aplikace a porovnání generovaných dat proti aplikaci SAR.

Klíčová slova: Atop, Monitoring, PostgreSQL, Testování, C, Statistiky

Abstract

The aim of this thesis is to extend the Atop application with server – client communication as the current application is to be enhanced by the possibility of output into a socket which will be used on the part of the client. The receiving part, the server, will receive the data from the socket or as a file by generated by Atop. Then the data will be decompressed and saved into a database.

The first part of the thesis deals with a description of the original version of the Atop application, its functionality and outcome. The second chapter contains the analysis of the original application, a description of used technologies for upgrading the current version and the description of development of a new server application. Next the implementation of individual parts is described here. The third chapter deals with a description of a data layer of the application, contains an analysis and proposal for implementation of the database. In the last chapter the process of testing of whole application is mention and comparison of generated data in comparison with the SAR application.

Keywords: Atop, Monitoring, PostgreSQL, Testing, C, Statistics

Seznam použitých zkratek a symbolů

ACID	–	Atomicity, Consistency, Isolation, Durability
ANSI	–	American National Standards Institute
SQL	–	Structured Query Language
DML	–	Data manipulation language
MVCC	–	Multiversion concurrency control
T-SQL	–	Transact Structured Query Language
PL/SQL	–	Procedural Language Structured Query Language
LZ77	–	Lempel-Ziv 77
RAM	–	Random-Access Memory
CPU	–	Central Processing Unit
HDD	–	Hard Disk Drive
CMD	–	Command

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Důvodem zvolení tématu byla možnost navrhnout a realizovat řešení, které bude dlouhodobě využíváno a dále rozvíjeno pro monitoring různých serverů, od databázových přes webové až po telefonní ústředny.

Tato práce by měla zefektivnit práci s generovánými daty, kdy nad těmito daty bude možné provádět různé analýzy. Především nad daty vytěžování fyzického nebo virtualního stroje uživatelskými procesy. Umožní tak v co nejmenším možném čase zjistit zda a co způsobilo například maximální zátěž procesoru nebo jak za určitou dobu rostly nároky na fyzickou paměť. Proto je také tento systém sledování vhodný i pro vývojáře různých aplikací. Takto upravený program Atop umožní při zdánlivě nenápadném růstu paměti, po delší dobu, zjistit například neuvolnění paměti sledované aplikace. Implementace by měla snížit náročnost zápisu statistik a potřebu fyzického přístupu k těmto statistikám, sníží se tak i vytížení monitorovaného stroje a jeho potřeb na datový prostor.

Tím, že jsou tyto statistiky ukládány do databáze je takto umožněna rychlejší práce nad vybranými daty. Při vyhledávání již zmíněného procesu, i po delší době, lze zjistit, jak proces pracoval v čase s procesorem, pamětí nebo jak často přistupoval na disk. V případě, že by byly statistiky ukládány na disk jako RAW soubor, nebylo by možné tato data tak účinně a rychle prohledávat.

2 Srovnání Atopu s jinými produkty

2.1 Top

Top je původní unixový nástroj pro zobrazení zátěže systému. Podobně jako Atop zobrazuje statistiku systému, informace o systému (doba běhu, počet přihlášených uživatelů, průměrná zátěž v různých časových intervalech, počet aktivních a běžících procesů, využití paměti, procesoru, swapu a seznam procesů). Také umí pracovat s procesy (zaslat signál SIGKILL, nastavovat různou prioritu). Na rozdíl od Atopu neumí zobrazovat statistiky jednotlivých síťových rozhraní, statistiky disků, statistiky ukládat do souboru a komprimovat je.

2.2 Htop

Výstup Htopu je ve srovnání s Top a Atopem graficky lépe zpracovaný, respektive více přehledný. Výstupní data se proti Topu nijak neliší, naproti tomu, Atop disponuje výpisem síťových rozhraní a jejich statistikami, také obsahuje podrobnější statistiky procesů. Funkcionalita je v dalších atributech stejná, jako u předchozích dvou monitorů systému.

2.3 Nmon

Nmon zobrazuje podobné statistiky jako Top a Atop, ale kromě toho nabízí statistiku obsazení disku nebo podrobnější pohled na počítač (označení procesoru, jádro, verzi Linuxu atp.). Ve srovnání s Atopem neumí tato data ukládat do souboru.

2.4 SAR

SAR stejně jako Atop zobrazuje statistiky systému. Na rozdíl od Atopu neumí zobrazit statistiky jednotlivých procesů.

2.5 IOTOP

Zobrazí pouze informace, jak který proces využívá disk. Zároveň vypíše přehled jakou rychlostí jsou data zapisována a čtena z pevného disku.

2.6 LATENCYTOP

Vypíše pouze informace s reakčními časy. Pomůže odhalit, co brzdí spouštění a běh programu. Může jít například o čekání na vstup z disku nebo ze sítě.

2.7 POWERTOP

Aplikace zjistí, které spuštěné procesy způsobují probouzení procesů a tím zvýšenou spotřebu energie. Ve srovnání s dalšími již zmíněnými aplikacemi, je tato aplikace jediná, která tuto informaci umí zobrazit, na rozdíl od ostatních nezobrazí žádné další statistiky.

2.8 Proč Atop

Atop byl nejvhodnější volbou z monitorovacích systémů. Jako jediný pokrývá nejvíce statistik zátěže systému. Díky použití standardních funkcí a dobře komentovaného zdrojového kódu je právě on vhodný pro různé úpravy.

```
ATOP - vmmre01                        2013/05/01  17:10:45                        -f-a-1---                        2s elapsed
PRC |  sys    0.01s |  user   0.01s |  #proc     72 |  #tslpi    79 |  #tslpu     0 |  #zombie    0 |  #exit    0/s  |
CPU |  sys       1% |  user      1% |  irq       0% |  idle     98% |  wait      0% |  curf 2.50GHz |  curscal   ?%  |
CPL |  avg1   0.00  |  avg5   0.00  |  avg15  0.00  |  csw    443/s |  intr   169/s |               |  numcpu     1  |
MEM |  tot  359.2M  |  free  13.8M  |  cache 145.7M |  buff   99.4M |  slab   49.1M |  shmem   0.0M |  shrss   0.0M  |
SWP |  tot  512.0M  |  free  453.5M |               |               |               |  vmcom 211.0M |  vmlim 691.6M  |
PAG |  scan    0/s  |  steal   0/s  |  stall   0/s  |               |               |  swin    0/s  |  swout    0/s  |
MDD |           md0 |  busy     0%  |  read    0/s  |  write   0/s  |  MBr/s  0.00  |  MBw/s   0.00  |  avio 0.00 ms  |
DSK |           sdb |  busy     0%  |  read    0/s  |  write   2/s  |  MBr/s  0.00  |  MBw/s   0.01  |  avio 1.33 ms  |
DSK |           sdc |  busy     0%  |  read    0/s  |  write   0/s  |  MBr/s  0.00  |  MBw/s   0.00  |  avio 0.00 ms  |
DSK |           sda |  busy     0%  |  read    0/s  |  write   0/s  |  MBr/s  0.00  |  MBw/s   0.00  |  avio 0.00 ms  |
NET |  transport    |  tcpi   1/s   |  tcpo    2/s  |  udpi    0/s  |  udpo    0/s  |  tcpao   0/s  |  tcppo    0/s  |
NET |  network      |  ipi    1/s   |  ipo     2/s  |  ipfrw   0/s  |  deliv   1/s  |  icmpi   0/s  |  icmpo    0/s  |
NET |  eth0    ---- |  pcki   3/s   |  pcko    2/s  |  si   2 Kbps  |  so   3 Kbps  |  erri    0/s  |  erro     0/s  |
NET |  lo      ---- |  pcki   0/s   |  pcko    0/s  |  si   0 Kbps  |  so   0 Kbps  |  erri    0/s  |  erro     0/s  |

  PID   TID  RUID      EUID       THR SYSCPU  USRCPU  VGROW   RGROW  RDDSK   WRDSK ST  EXC S  CPUNR  CPU  CMD          1/4
10701     -  root      root         1 0.01s   0.01s     0K      0K     0K      0K --    - R      0   1%  atop
 3230     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  python
10145     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  atop
 3165     -  named     named        4 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  named
19089     -  root      root         2 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  clusterstat.py
10676     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  sshd
10678     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  bash
29466     -  root      root         2 0.00s   0.00s     0K      0K     0K      0K --    - R      0   0%  python
28078     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - R      0   0%  python
29683     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - R      0   0%  python
18865     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  python
 3083     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  syslog-ng
10182     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  sar
10187     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  sadc
 3050     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  dhclient
 8811     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  sshd
 3263     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  crond
 3297     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  atd
    1     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  init
19120     -  root      root         1 0.00s   0.00s     0K      0K     0K      0K --    - S      0   0%  screen
```

Obrázek 1: Atop

3 Detailní popis programu atop

Program ATOP je interaktivní monitor zobrazující zatížení systému Linux. Ukazuje zatížení nejkritičtějších hardwarových zdrojů (z pohledu výkonu) na úrovni systému, tj. cpu, paměť, disk, síť a množství interních informací o jádře OS (zpracování přerušení, přepínání kontextu, informace o scheduleru, a další).

ATOP ukazuje, které procesy jsou zodpovědné za indikované zatížení s ohledem na procesor a paměti zatížené na procesní úrovni. Diskové zatížení je zobrazeno, pokud proces "zápisu dat"je aktivní v jádře, nebo v případě že je nainstalován jaderný patch "cnt". Zatížení sítě se zobrazí pouze na proces. Za každý interval (výchozí: 10 sekund) atop zobrazuje vždy využité prostředky na úrovni systému (CPU, paměť, disky a síťové vrstvy), následuje seznam procesů, které byly aktivní při posledním intervalu, všechny procesy, které byly beze změny během posledního intervalu nejsou zobrazeny. Pokud seznam aktivních procesů není celý na obrazovce, je zobrazen jen vrchol seznamu (seřazen v pořadí aktivity) v tomto seznamu je možné scrollovat a měnit seřazení.

Intervaly jsou opakovány, dokud není dosaženo počtu vzorků uvedených jako argument příkazu, nebo dokud není v interaktivním režimu stisknuta klávesa "q".Když se ATOP spustí, kontroluje, zda je směrován výstupní kanál na obrazovku nebo do souboru. V prvním případě se výstup na obrazovce řídí přes ncurses, tzn. využití knihovny poskytující rozhraní pro tvorbu aplikací v textovém režimu běžících v terminálu. V druhém případě se vytváří ASCII-výstup.V interaktivním režimu, se výstup ATOPU mění dynamicky podle současných rozměrů obrazovky / okna. Lze jej také ovládat stiskem kláves.Je však také možné stanovit takovou klávesu jako parametr v příkazovém řádku. V druhém případě ATOP přepne do určeného režimu, tento režim může být změněn opět interaktivně. Určení parametru je zvláště užitečné, když běží ATOP s výstupem do souboru. Použité příznaky jsou stejné jako klávesy, které mohou být použité v interaktivním režimu.

3.1 Proces zápisu

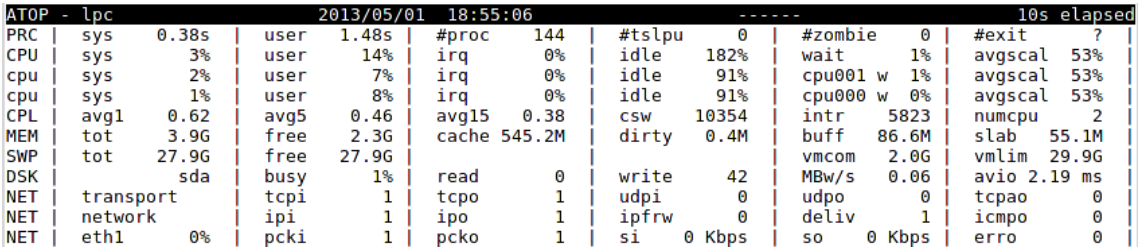
Aby bylo možné uchovávat statistiky systému a procesů pro dlouhodobou analýzu např. pro kontrolu zatížení systému a zatížení jednotlivými procesy spuštěnými např. včera 03:00 - 16:00, je možné ukládat systémové statistiky na úrovni procesu v komprimovaném binárním formátu. Pokud tento soubor již existuje a je uznán Atopem jako platný soubor, bude zapisovat nové vzorky na konec souboru, pokud soubor neexistuje, bude vytvořen. Ve výchozím nastavení zapisuje pouze procesy, které byly v aktivním stavu. Takto uložený soubor lze číst a vizualizovat.

3.2 Popis výstupu

V záhlaví se zobrazuje čas a nastavená délka tohoto vzorku. Pro každý vzorek, Atop nejprve zobrazí řádky vztahující se k činnosti na úrovni systému. Pokud daný systémový zdroj nebyl použit během intervalu, je potlačen celý řádek vztahující se k tomuto prostředku. Takže počty řádků na úrovni systému se mohou lišit pro každý vzorek. Poté

se zobrazí seznam procesů, které byly aktivní při posledním intervalu. Tento seznam ve výchozím nastavení je seřazen podle vytížení cpu, nastavení může být změněno pomocí kláves.

4 Systémové prostředky



Obrázek 2: Systémové prostředky

Informace na úrovni systému se skládají z následujících výstupních údajů:

4.1 Součty za procesy a vlákna - PRC

Tento řádek obsahuje celkový procesorový čas, spotřebovaný v systémovém režimu a v uživatelském režimu, celkový počet procesů přítomných v této chvíli, počet zombie procesů a počet procesů, které skončily během intervalu.

4.2 Vytížení CPU - CPU

Alespoň jeden řádek je zobrazen po celkovém obsazení všech CPU dohromady. V případě multi-procesorových systému, je každý další řádek zobrazen pro každý jednotlivý procesor, kde jsou seřazeny podle vytížení. Neaktivní CPU ve výchozím nastavení nejsou zobrazeny. Řádky ukazující využití cpu obsahují v posledním poli její číslo. Každý řádek obsahuje procenta procesorového času stráveného v režimu jádra všemi aktivními procesy (”sys”), procenta procesorového času spotřebovává v uživatelském režimu (”uživatel”), pro všechny aktivní procesy (včetně procesů probíhajících s hodnotami vyššími než nula), procento procesorového času stráveného zpracováním přerušení (”irq”), procento nevyužitého času procesoru (”idle”), procento času stráveného čekáním na diskové operace (”wait”). Poslední sloupec zobrazuje počet CPU a čekání v procentech(”w”) pro daný procesor. Počet řádků zachycujících využití cpu může být omezen. Při spuštění Atopu na virtuálním stroji, může zobrazit procesorový čas jiných virtualních strojů běžících na stejném hardwaru. Takový čas je potom zobrazen jako (“steal”). Využití procesorového času fyzickým strojem je zobrazeno jako (“host”).

4.3 CPU load information - CPL

Tento řádek obsahuje údaje vytížení odrážející počet procesů, které čekají na CPU, nebo čekajících na diskové operace. Tyto údaje jsou v průměru za 1 (”avg1”), 5 (”avg5”) a 15 (”avg15”) minut. Dále je zobrazen počet přepnutí kontextu (”CSW”), počet obsluhovaných přerušení (”intr”) a počet cpu, které jsou k dispozici. Pokud šířka terminálu neumožňuje všechny tyto položký vypsat, je zobrazena jen relevantní část.

4.4 Využití paměti - MEM

Tento řádek obsahuje celkové množství fyzické paměti (”tot”), množství paměti, která je v současné době nevyužita (”free”), množství paměti v použití jako cache stránek (”Cache”), velikost paměti v rámci stránek mezipaměti, které musí být zapsány na disk (”dirty”), množství použité paměti pro meta data souborového systému (”buff”) a množství paměti používané pro malloc.

4.5 Swap - SWP

Tento řádek obsahuje celkové množství odkládacího prostoru na disku (”tot”) a množství volného odkládacího prostoru (”free”).

4.6 Paging frekvence - PAG

Tento řádek obsahuje počet procesů, které jádro OS skenovalo ve snaze uvolnit paměť (pokud byla překročena daná hranice pro znovuuvolňování (reclaim) (”scaning”), dále počet případů, kdy jádro muselo uvolňovat paměťkvůli urgentnímu nedostatku (”stall”). Rovněž počet paměťových stránek čtených z odkládacího prostoru (”swin”) a počet paměťových stránek, které systém zapsal do odkládacího prostoru (”swout ’).

4.7 LVM / MDD / DSK - využití fyzických a logických disků

Zde jsou vyobrazeny informace o diskových jednotkách, které zobrazují pro každou jednotku tyto atributy:

- část času, po který byl disk zaneprázdněn vyřizováním žádostí (”busy”)
- počet žadostí o čtení (”read”)
- počet žádostí o zápis (”write”)
- počet KiB za čtení (”KB/r ’)
- počet KiB za zápis(”KB / w’)
- propustnost čtení (’MBr/s”)
- propustnost zápisu (’MBW/s”)
- průměrná hloubka fronty (”AVQ”)
- průměrný počet milisekund potřebných pro vyhledávání (”Avio”)

4.8 NET - využití rozhraní (TCP/IP)

První řádek je vyhrazen pro činnost transportní vrstvy (TCP a UDP).

4.8.1 Zde jsou zobrazeny tyto atributy:

- počet přijatých segmentů (”tcpi”)
- počet odeslaných segmentů TCP (”tcpo”)
- počet přijatých UDP datagramů (”udpi”)
- počet odeslaných UDP datagramů (”udpo”)
- počet aktivních TCP (”tcpao”)
- počet pasivní TCP (”tcppo”)
- počet výstupních přenosů TCP (”tcprs”)
- počet vstupních chyb TCP (”tcpie ”)
- počet výstupních přenosů TCP (”tcpor ”)
- počet UDP žádáných portů (”udpnp ”)
- počet vstupních chyb UDP (”tcpie ’)

4.8.2 Druhý řádek slouží pro IP vrstvu a zobrazuje:

- počet datagramů protokolů vyšší vrstvy využitých pro přenos (”IPO”)
- počet datagramů obdržených omylem (”IPP”)
- počet přijatých datagramů, které byly předány do jiných rozhraní (”ipfrw”)
- počet datagramů, které byly dodány na vyšší vrstvě protokolů (”deliv”)
- počet přijatých ICMP datagramů (”icmpi”)
- počet odeslaných ICMP datagramů (”icmpo”)

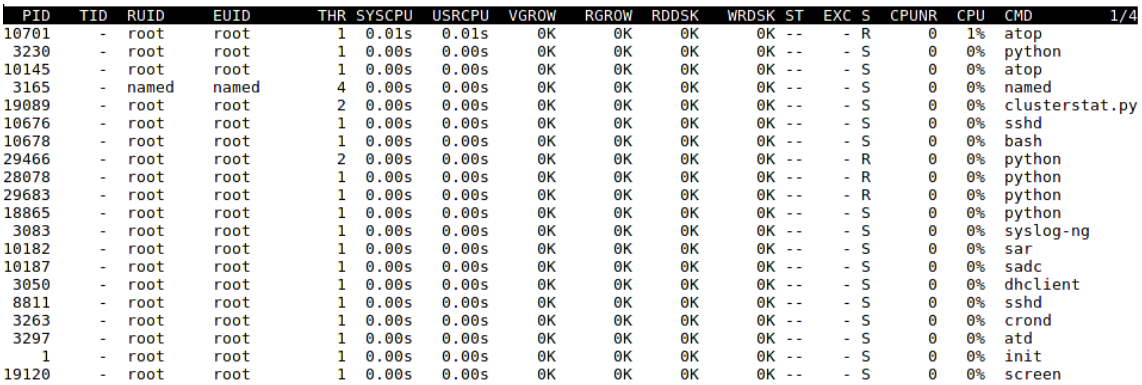
Další řádky jsou určeny pro aktivní rozhraní seřazené podle činnosti.

4.8.3 Další atributy aktivních síťových rozhraní:

- počet přijatých paketů (”pcki”)
- počet přenesených paketů (”pcko”)
- efektivní množství bitů přijatýchza sekundu (”si”)
- efektivní množství bitů přenesených zasekundu (”co”)
- počet kolizí (”Sb”)

- počet přijatých paketů vícesměrového vysílání (”mlti”)
- počet chyb při příjmu paketu (”ERRI”)
- počet chyb při přenosu paketu (”erro”)
- pokles přijatých paketů (”drpi”)
- pokles přenesených paketů (”drpo”).

5 Statistiky jednotlivých procesů



PID	TID	RUID	EUID	THR	SYSCPU	USRCPU	VGROW	RGROW	RDDSK	WRDSK	ST	EXC	S	CPUNR	CPU	CMD	1/4
10701	-	root	root	1	0.01s	0.01s	0K	0K	0K	0K	--	-	R	0	1%	atop	
3230	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	python	
10145	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	atop	
3165	-	named	named	4	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	named	
19089	-	root	root	2	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	clusterstat.py	
10676	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	sshd	
10678	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	bash	
29466	-	root	root	2	0.00s	0.00s	0K	0K	0K	0K	--	-	R	0	0%	python	
28078	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	R	0	0%	python	
29683	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	R	0	0%	python	
18865	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	python	
3083	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	syslog-ng	
10182	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	sar	
10187	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	sadc	
3050	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	dhclient	
8811	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	sshd	
3263	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	crond	
3297	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	atd	
1	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	init	
19120	-	root	root	1	0.00s	0.00s	0K	0K	0K	0K	--	-	S	0	0%	screen	

Obrázek 3: Statistiky jednotlivých procesů

V návaznosti na informace na úrovni systému jsou v jednotlivých intervalech zobrazeny procesy a jimi využité zdroje. Zde jsou zobrazeny využití paměti, síťové požadavky a využití procesorového času.

5.1 Popis výstupů:

5.1.1 CPU

Zde Atop zobrazuje využití CPU procesem, jako procento využití celkové kapacity procesoru. Je zde zobrazena také identifikace procesoru a hlavního vlákna CPUNR. SYSCPU, která vyjadřuje časovou náročnost procesu v sekundách.

5.1.2 CMD

Název procesu CMD. Tento název může být obklopen "menší/větší než", což znamená, že tento proces byl dokončen v posledním intervalu. Za zkratkou "CMD"v záhlaví řádku, se zobrazí aktuální číslo stránky a celkový počet stránek výpisu. Zobrazením COMMANDLINE odpovídá celému příkazu procesu včetně jeho argumentů předaných na příkazové řádce. Filesystem groupid a userid je zobrazeno za FSGID a FSUID, počet major/minor výpadků stránek procesu Atop vypisuje jako MAJFLT a MINFLT.

5.1.3 DISK

Průměrná velikost jednoho čtení procesem z disku je zobrazena jako AVGRSZ, zápisu pak AVGWSZ, DSK procentualní podíl využití disků procesem, tzn. počet přístupů v posledním intervalu v procentech. Zápis fyzických dat na disk WRDSK, čtení RRDSK. WCANCEL zobrazí odstraněná data procesu.

5.1.4 PROCES – hlavní informace

Datum, kdy byl proces dokončen je vyjádřen jako ENDATE, čas ENTIME, v případě, že proces stále běží Atop zobrazí „active“. Efektivní uživatelské id, které proces obdržel je vypsáno jako Euid, EXC je návratový kód ukončeného procesu. Priorita procesů NICE je vyjádřena na stupnici od -20 (vysoká priorita) až po 19 (nízká priorita). Atop vypíše na výstup také počet aktivních a ukončených procesů NPROCS, PID id procesu, POLI politika procesu (normal, batch, idle, fifo, round robin). Každý paralelní proces má svůj PPID, tedy pid podprocesu. Priorita procesů je zobrazena jako PRI, real-time procesy mají PRI v rozmezí 0 – 99 a timesharing 100 - 139. Realné id skupiny ve kterých je proces spuštěn jsou zobrazeny jako RGID. Současný stav hlavního vlákna procesu zobrazuje jeden z těchto znaků: „R“ v současné době zpracovává, „S“ čeká na událost, „D“ uspaný, „Z“ zombie proces, „T“ pozastaven, „W“ swap, „E“ ukončení. Stav procesů ST vypíše, kdy byl proces zahájen, když byl proces zahájen v počátečním intervalu je výpis „N”, „E” pak byl dokončen v intervalu. Celkový počet vláken zobrazuje sloupec THR.

5.1.5 MEM

Využití paměti zobrazí jako MEM což je procento týkající se využití volných prostředků procesem. Množství spotřebované virtualní paměti VGROW, růst tohoto atributu může být způsoben např. požadavkem malloc() nebo připojením segmentu sdílené paměti. Virtuální růst může být také negativní tím, že je např. voláno free() nebo odejmut segment sdílené paměti. Velikost textového segmentu je označena jako VSTEXT. Množství využité rezidentní pamětí je zobrazeno za atributem RGROW, celkové využití paměti v kilobytech nebo megabytech pak Rsize.

5.1.6 NET - Síť

NET je podíl tohoto procesu týkající se celkového zatížení sítě. RNET/SNET je počet TCP a UDP paketů přijatých/přenesených tímto procesem. Počet přijatých datagramů zobrazených ve sloupci RAWRC a odeslaných jako RAWSND. RDDSK zobrazí počet přístupů k fyzickému souboru. Průměrná velikost přijímaného TCP spojení je vyjádřena za TCPRASZ, počet žádostí je zobrazen ve sloupci nazvaném TCPRCV, velikost přenášeného bufferu TCPSASZ. TCPSND znamená počet odesláných žádostí tímto procesem pro TCP socket. SUDPRASZ je průměrná velikost přijímaného a UDPSASZ přeneseného UDP paketu v bajtech. UDPRCV je počet přijatých a UDPSND počet odeslaných požadavků vydaných tímto procesem pro socket UDP. Sloupec USRCPU vyjadřuje spotřebu procesorového času v uživatelském režimu, v důsledku zpracování vlastního textu programu.

6 Použité technologie

6.1 Programovací jazyk C

Jedná se o nízkoúrovňový, kompilovaný programovací jazyk, který vyvinuli Ken Thompson a Dennis Ritchie pro potřeby operačního systému Unix. Dnes se používa ANSI standard C99,který implementuje různá rozšíření jazyka například:

- Inline funkce
- Deklarace proměných je možná kdekoliv
- Nové datové typy long long int, bool
- Pole nekonstantní velikosti
- Komentovaní řádku jako v jazyce C++
- Nové knihovny, funkce, hlavičkové soubory

6.2 Netbeans

Open source vývojové prostředí, pro volitelné platformy C/C++, JAVA, Python, PHP, na kterém, má velký podíl firma Sun Microsystem, slouží pro vývoj softwarových aplikací různého typu např. konzolových, webových apod.

6.3 Kate

Jedná se o jednoduchý textový editor s podporou syntaxe C/C++, PHP a dalších. Kate nabízí také implementovaný shell.

6.4 GCC

GNU Compiler Collection je volně dostupná sada překladačů, poprvé vytvořená v roce 1985 Richardem Stallmanem v jazyce Pastel. V roce 1987 byla přepsána do jazyka C Lenem Towerem a Richardem Stallmanem. Od roku 1994 je překladač uznán jako implicitní pro distribuce Unixu.

6.5 Valgrind

Nástroj pro ladění a profilování spustitelných souborů, distribuován pod licencí GNU. Detekuje chyby související se správou paměti, zachycuje všechny volání funkcí malloc/free/new/delete. Jeho výstupem jsou textové řetězce, které upřesňují chybové výpisy.

6.6 PostgreSQL

Je plnohodnotný, spolehlivý a bezpečný relační databázový systém vyvíjený více jak patnáct let. Je funkční na všech dostupných a rozšířených systémech Linux, Mac OS X, Solaris a Windows. Splňuje všechny podmínky ACID, podporuje cízí klíče, operace join, pohledy, funkce, procedury, triggery. Obsahuje SQL 2 a SQL 3 datové typy např. integer, numeric, boolean, char, varchar, date, interval a timestamp. Tento databázový systém je srovnatelný se všemi dostupnými komerčními systémy například PL/SQL, T-SQL.

Hlavní vlastnosti PostgreSQL jsou :

- Funkce - umožňuje spuštění bloku kódu
- Indexy - obsahuje podporu pro základní typ indexu B-tree, R-tree optimalizovaný pro geometrická data, hash, zobecněný vyhledávací strom GisT
- Triggery - události spouštěné nad akcemi z SQL DML příkaz.
- MVCC – zajištění konzistence dat databázového systému.
- Uživatelem definované objekty
- Dědičnost

6.7 pgAdmin

PgAdmin je multiplatformní, administrační, open source rozhraní pro správu databáze PostgreSQL a databázi odvozených jako například Greenplum Database.

6.7.1 Přístup k datům

- K vybraným datům se lze dostat pomocí dotazovacího nástroje. Tento nástroj má zvýrazňování syntaxe a je v něm obsažena velmi rychlá datová mřížka sloužící k zobrazení, zadávání a správě dat.

6.7.2 Přístup k objektům

- Objekty jsou zobrazovány vč. jejich definice v SQL a seznamem vlastností. Může se také zobrazit seznam závislostí závisejících objektů a statistiky.

7 Implementace rozšíření server – klient

Monitor systému Atop umožňuje ukládat generované statistiky do souboru. Tento způsob však není vhodný pro stroje, na kterých je potřeba minimalizovat zbytečnou zátěž systému. Přikladem takového stroje mohou být např. telefonní ústředny. Je zde nutné omezit činnost Atopu k tomu, že odešle generované a komprimované struktury na server, který tato data zpracuje. Podpora je od verze 2.0 a vyšší. Struktury dále převede do binární podoby a odešle do databáze. V rámci kapitoly implementace rozšíření server – klient, jsou zmíněna rozšíření, která splňují všechny dané požadavky.

7.1 Analýza původní aplikace

Původní aplikace ukládá statistiky pouze do souboru, tzn., že uživatel může zapisovat statistiky pouze na disk a to může vést k nechtěné zátěži systému a problému udržet tato data dlouhodobě s ohledem na volné zdroje. Proto byla nutnost přidat k původní aplikaci funkci, která umožní rozšíření o odeslání těchto dat na socket, což je jeden z prostředků meziprocesní komunikace, ale liší se především tím, že komunikující procesy nemusí být na stejném stroji. Zpracování probíhá na jiném stroji, kde není nutnost striktně hlídat jakoukoliv generovanou zátěž aplikacemi a volné prostředky. Odpadá tak zbytečný přístup na sledovaný stroj jen proto, aby si uživatel prošel nebo přesunul generované statistiky. Tento problém řeší druhá část projektu server.

7.2 Analýza implementace serveru

Vlastnosti, které vyplynuly z analýzy původní aplikace, bylo potřeba doplnit po implementaci zápisu na straně klienta a o čtení na straně serveru. Tzn. přečíst data ze socketu. Přečtená data bylo nutné dekomprimovat, kontrolovat správnost zapsaných dat (podle verze a přiděleného kontrolního čísla), ukládat do vhodných struktur, tyto struktury procházet a převádět do binární podoby, aby je bylo možné odeslat do databáze. Pro další potřeby byla nutnost umět zpracovat generovaný soubor Atopu, resp. číst soubor, dekoprimovat jej převádět na jednotlivé struktury a již zmíněným způsobem odeslat do databáze.

7.3 Zapis na socket - klient

Jak již vyplynulo z analýzy, bylo potřeba, aby původní aplikace atop zapisovala na socket, toho se dosáhlo tak, že se z původního nezkompilovaného zdrojového kódu zápisu do souboru použily nutné funkce a struktury. Struktury rawheader, která obsahuje další struktury nutné pro práci s přijatými daty, především proměnnou sstatlen a tstatlen, tyto proměnné nesou informaci o délce struktur sstat a tstat. Do struktury tstat jsou ukládány pouze statistiky s procesy z kapitoly výstup procesy, Další strukturou je sstat tato struktura nese data o systémových prostředcích, to je zmíněno v kapitole systémové prostředky. Tyto dvě struktury jsou jako jediné komprimované, to z důvodu množství

dat v nich obsažených. Kompresi těchto dvou struktur řeší funkce compress, která je implementováná v knihovně Zlib a obalená ve funkci rawwrite.

Zápis na socket vychází z původní funkce zápisu do souboru rawwrite s tím rozdílem, že se neotevírá soubor a zapisuje do něj, ale zápis probíha na daný socket. Zápis na socket umožňuje stejně jako v původní implementaci funkce write, ale za první parameter je dosazen místo file descriptoru na soubor descriptor na již otevřený socket. To se dosáhlo tím, že se doimplementovaly další funkce vhodné pro práci se sockety, hlavní funkcí pro práci se socketem je funce socket, tato funkce vytvoří soket a specifikuje komunikační styl, který by měl být jedním z uvedených stylů:

- SOCKSTREAM - přenáší data spolehlivě mezi více vzdálenými sockety.
- SOCKDGRAM - přenáší individualně řešené pakety. Tento styl přenášení paketů je přenosově nespolehlivý.
- SOCKRAW - tento styl poskytuje přístup k nižším síťovým protokolům a rozhraním.

Další důležitou funkcí je connect, která inicializuje spojení ze socketu na socket, jehož adresa je uvedena v předem nastavené struktuře sockaddr dalším argumentem je velikost sockaddr. Funkce connect čeká na dané adrese dokud server neodpoví na žádost. Běžná návratová hodnota je 0. Pokud dojde k chybě připojení, vrátí hodnotu -1. Struktura sockaddr obsahuje potřebné proměnné pro připojení, které jsou vyplněny adresou, socketem a typem připojení. Tyto funkce a struktury jsou obsaženy v knihovně sys/socket.

K vyvolání zápisu na socket je potřeba před spuštěním aplikace vyplnit dva paramatery „-p“ (za který je dosazen port) a parametr „-a“ (za který je dosazena ip adresa stroje na který chceme data odesílat) tzn. adresa serveru. Před tyto dva parametry je potřeba dosadit parametr „-N“, který slouží ke spuštění funkce zápisu dat na socket. Poté čeká již zmíněná funkce na připojení k serveru. Do této doby se v konzoli nic nezobrazí. Po připojení se kurzor přesune v konzoli na nejvyšší možnou pozici a vypíše „Connect to server“. Další výpis zde není potřebný z důvodu, že tento klient poběží především jako daemon.

8 Atop a komprese dat na straně klienta

Atop komprimuje data pomocí standardizované knihovny Zlib, která využívá DEFLATE algoritmus. Jedná se o kombinaci dvou bezztrátových algoritmů LZ77 a Huffmanova kódování. Tato data je možné ukládat do souboru nebo zapisovat na socket.

8.1 LZ77

Tento algoritmus nahrazuje opakující se znaky, které se již dříve v datech objevily za odkazy. Každý odkaz je složen ze vzdálenosti, ve kterém se skupina znaků nachází a délky skupiny znaků. Když je odkaz kratší jako reprezentace skupiny znaků dochází k úspoře místa.

8.2 Huffmanovo kódování

Znaky se konvertují do vstupního souboru bitových řetězců různé délky. Ty znaky, které se vyskytují nejčastěji, jsou konvertovány do bitových řetězců s nejkratší délkou. Znaky vyskytující se velmi zřídka, jsou konvertovány do delších řetězců. To probíha ve dvou fázích. V první fázi se prochází soubor a vytváří statistiku četnosti znaků a v druhé fázi se vytváří za využití statistik binární strom a komprese dat.

8.3 Komprese dat před uložením do databáze

Server přijme surová data, která dekomprimuje pomocí knihovny Zlib a prochází je, ty potom ukládá do databáze. Zde nastává problém s množstvím dat, především s těmi, které obsahují struktury procesů. Oproti strukturám se systémovými prostředky, jako procesorový čas, obsazená paměť, swap, je struktur s informacemi o procesech tolik, jako spuštěných procesů. Proto předtím než se data nabufferují do paměti a zapíšou do databáze, jednoduchý algoritmus je projde a vypustí všechny ty, co obsahují nulové řádky.

9 Implementace serveru

Z analýzy vyplynula nutnost implementace druhé části projektu dle zadaných požadavků, tzn. část serveru, která zapíše data ze socketu do databáze a druhou část, která čte data ze souboru a zapisuje je do databáze.

9.1 Zápis dat ze souboru do databáze

Pro případ, kdy je potřeba číst data ze souboru generovaného Atopem a ukládat tato data do databáze, bylo nutné implementovat čtení ze souboru a vkládání těchto dat do struktur. Toho se docílilo tak, že po přečtení parametru na vstupu, který se skládá ze jména souboru a cesty, se tento parametr předá funkci „open“, která využívá proud bytů. Této funkce je využito proto, aby nebylo nutné soubor udržovat v paměti jako celek a zabírat, tak zbytečně rozsáhlý prostor. V některých případech by mohl mít výstupní soubor Atopu až stovky megabytů. Funkce ”lseek”vytvoří pozici v souboru, která je určena deskriptorem souboru fd, na místě určeném posunutím o offset. Následující čtení souboru se uskuteční na této pozici. Pozice je určena vždy, tak aby čtení začalo hlavičkou a skončilo komprimovanými daty procesů, tato data spolu se statistikami systému jsou dekoprimována funkcí „uncompress“, která stejně jako funkce compress vychází ze stejné knihovny Zlib. Tyto statistiky jsou uloženy v paměti jako struktury, až do doby, kdy jsou data odeslána do databáze a paměť je uvolněná. V případě, že některá data neodpovídají podporované verzi Atop 2.0 a vyšší je na standardní výstup vypsána chybová odpověď. Zápis dat ze souboru do databáze, se vyvolá parametrem „-f“ po tomto parametru se zapíše cesta k souboru.

9.2 Zapis dat ze socketu

Pro zápis dat odeslaných na socket klientem do databáze, je nutné nastavit a otevřít socket stejně jako na straně klienta, s tím rozdílem, že strana serveru musí přijmout spojení. Toho se dosáhlo použitím funkce „listen“, která uvede socket do stavu kdy čeká na žádost o spojení „listening“. V případě, že je socket ve stavu „listening“ volá se funkce accept, která vrátí parametry prvního spojení ve frontě a nový socket, který slouží ke komunikaci se stranou klienta. Původní socket se tak může znovu použít pro čekání na další spojení. Proto, aby bylo možné komunikovat asynchronně s více než jedním klientem, má strana serveru implementovanou asynchronní komunikaci pomocí funkce ”select”, která umožní čekat událost na socketu, jako je zápis na socket ze strany klienta. Potom co je událost přijata se uloží descriptor připojeného klienta. Na tomto descriptoru se provádí čtení, tak dlouho dokud se klient neodpojí a jeho descriptor je zrušen. Zrušení descriptoru a odpojení klienta proběhne i v případě, že přijatá data jsou nečitelná. Aby nedošlo k dalšímu nechtěnému čtení, které by mohlo spuštěnou aplikaci zpomalit, nebo ovlivnit její jinak bezchybný chod.

O čtení z descriptoru se stará funkce „read“, server jako první přečte hlavičku a velikosti dalších dvou struktur, alokuje potřebnou paměť. Pokud nejsou potřebná data závadná souhlasí hlavička a kontrolní číslo, pak je provedeno čtení dalších dvou struktur,

které nesou data o procesech a systému. Předtím, než server přistoupí k dekomprimaci dat a převedení na struktury, je porovonávána velikost. V případě, že velikost souhlasí s danou velikostí předanou v hlavičce, je s daty provedena dekomprimace a převedení na danou strukturu. Po převedení těchto dat do struktur jsou data odeslána do databáze. Čas vzorku spolu se jménem stroje je uložen v paměti, tak aby bylo možné zjistit čas jednotlivých vzorků a jméno stroje ze kterého byla data generována. Čtení na socketu je možné vyvolat parametrem „-p“, který je následován číslem portu ze kterého jsou data čtena.

10 Práce s daty a převod do binární podoby

Poté co jsou data přečtena ze socketu nebo souboru jsou tato data, převáděna do binární podoby pomocí jednotlivých funkcí. Tyto funkce převádí jednotlivé datové typy, nebo zapisuje do paměti dané příkazy jako je vytvoření řádku. Data jsou serializována tak, aby souhlasila s délkou datového typu v databázi. Těchto funkcí je celkem osm pro nejpoužívanější datové typy:

- build_null - serializuje v určené části paměti nulu, která se projeví v databázi jako NULL.
- build_char - serializace datového typu char v databázi se projeví jako datový typ char.
- build_int - serializace datového typu integer, tento datový typ je do databáze zapsán jako integer, lze použít pro datový typ smallint, integer, serial.
- build_varchar - tento datový typ převádí řetězec znaků na varchar, ale lze jej použít také pro datový typ, který je v PostgreSQL určen jako text.
- build_bigint - tato funkce převede datový typ long long int na bigint .
- build_float - převádí datový typ float na float, tak aby byl čitelný pro databáze.
- build_column - tato funkce vytváří daný počet sloupců, počet je určen parametrem.
- build_timestamp - převede strukturu time_t na datový typ timestamp.

Tyto funkce jsou obaleny dalšími funkcemi, které jsou určeny pro ověření správnosti dat a provedení již zmíněné komprimace, před uložením do databáze nad danou strukturou. Zajišťují také uvolnění paměti v případě, kdy nejsou data dále potřebná a jsou přijata databázi. Zde se také rozhodne, jak dlouho mají být data bufferována.

10.1 Zápis dat do databáze

Ještě předtím než se data budou převádět do binárního formátu, vytvoří funkce „makeraw“ strukturu a místo v paměti pro určitá data. To je však podmíněno vypočítáním velikosti vybraných dat, u kterých je však nutné počítat s jistým rozdílem ve velikosti datových typů v databázi a programovacím jazyce. Následně je do této struktury uložen příkaz, který se má provést nad databází, v případě ukládání přijatých dat, je to příkaz „COPY TO“ v binárni formě. Při odesílání dat do databáze je nutné vytvořit neblokující čekání, aby bylo možné provádět další operace, ty obstará funkce poll, která stejně jako funkce select čeká na určitém file descriptoru, tak dlouho dokud nevrátí pollin. Tzn. že jsou připravena nepřečtená data. V případě, že PostgreSQL vrátí PGRES_COPY_IN, databáze může přijmout binární data, která mají striktně určenou strukturu.

10.2 Struktura binárnich dat

10.2.1 Signatura

Signatura je podpis, 11-bytová sekvence ve tvaru „PGCOPY n 377 r n 0“ - nulový bajt je povinnou součástí podpisu. Podpis je navržen tak, aby umožnil snadnou identifikaci souborů.

10.2.2 Hlavička

Udává řádek v záhlaví s názvy jednotlivých sloupců v souboru. Na výstupu, obsahuje první řádek názvy sloupců a na vstupu je tento řádek ignorován.

10.3 Flag

Flag je 32 bitů dlouhá bitová maska pro označení důležitých aspektů formátu. Bity jsou číslovány od 0 do 31. Toto pole je uloženo v síťovém pořadí bajtů (nejvýznamnější byte je první), stejně jako všechny celočíselné pole používaná ve formátu. Bity 16 až 31 jsou vyhrazeny pro označení kritických problémů formátu souboru. Strana která čte, by se měla odpojit, pokud zjistí, že je bit nastaven v tomto rozsahu. Bity 0 až 15 jsou vyhrazeny pro signalizaci. Pro správnou signalizaci musí být nastaven první bit na jedna a ostatní bity na nula.

10.3.1 Data

Obsahuje data převedená do binární nebo textové podoby. V případě, že je zvolena binární podoba dat, tak jako u tohoto projektu, je zde sice menší přenositelnost mezi verzemi PostgreSQL, ale čtení těchto dat je rychlejší.

10.3.2 End

Ukončí binární soubor, a připraví soubor k předání, poté se zavoláním funkce „PqputCopyEnd“ dokončí formátování dat.

10.4 Verifikace dat databází

V případě, že byla data předána databázi, vráti funkce „PqresultStatus“ stav ve kterém databáze skončila. Jestli zápis dat proběhl v pořádku vrací hodnotu „PGRES_COMMAND_OK“.

Když je databáze v jakémkoliv jiném stavu než v korektním, vrátí databáze chybu, provede se „rollback“ a tato chyba je vypsána funkcí „PqerrorMessage“. V případě, kdy je proveden „rollback“ nejsou žádna data uložena v databázi.

10.5 Bufferování dat

Statistiky procesů je možné uložit do databáze hned po přečtení. Těchto statistik je tolik jako aktivních procesů, tj. ve většině případech více než jeden. Proto není nutné data udržovat v paměti delší dobu. Statistiky celkových systémových prostředků je nutné udržovat v paměti delší dobu a hromadit je tak dlouho, dokud není jejich velikost dostatečná, aby se využila výhoda rychlosti přesunutí většího množství dat pomocí funkce „COPY TO“ do databáze.

11 Návrh databáze

11.1 Analýza požadavků na databázi

Hlavním požadavkem na databázi je dodržení základních pravidel pro správné a efektivní vytvoření databáze. Je nutné se vyvarovat jakékoliv redundancí dat. Vytvořit databázi takovým způsobem, aby obsahovala pouze atomické hodnoty, aby byly neklíčové atributy závislé na primárním klíči a neklíčové atributy byly vzájemně nezávislé. Na následujících řádcích je provedena analýza nejstěženějších entit.

11.2 Statistika systému

11.2.1 Procesor

Jedna z nejdůležitějších entit je cpustat, tj. celková statistika procesoru. Ta se skládá z různých atributů, jako jsou např. počet procesorů, jader, počet aktivních procesorů a dalších atribut, které jsou popsány v předešlé kapitole popis výstupu. Je nutné také zahrnout čas předešlého zápisu a čas aktualní, díky tomu lze dosáhnout výpočtu jednotlivých rozdílů. Dále je nutné určit, ze kterého stroje je daný zápis, proto je určeno jméno stroje. Různé další atributy jsou zahrnuty v další entitě, tato entita se nazývá cpustat, to proto aby se neporušilo jedno z pravidel správného vytvoření databáze atomičnosti.

11.2.2 Jednotlivá jádra

Jak již vyplynulo z předešlé kapitoly,, aby se neporušila atomičnost je důležité mít určité atributy v jiné entitě s názvem percpu z důvodu, že jakékoliv jádro může mít více vlastností např. procesorový čas nebo frekvenci.

11.2.3 Paměť

Tato entita s názvem memstat, je důležitou součástí hlavních statistik, jsou zde takové atributy jako např. fyzická paměť, volná paměť, informace o swapovacím prostoru. Zde je nutný čas aktuálního i předešlého vzorku a název stroje, který tato data produkoval.

11.2.4 Disky

Entita nesoucí název dskstat nese data s informacemi o discích fyzických, virtuálních, stroji z kterého tato data pocházejí a časy vzorků. Tato entita je rozšířená další entitou pro podrobnější informace o discích. Entita se nazývá perdsk, nese jmeno disku, stroje, počet zápisů a čtení. Z důvodu malého množství informace je možné tyto dvě entity sloučit aniž by byla porušena atomičnost databáze a tuto novou entitu pojmenovat pouze dsk.

11.2.5 Interface

Zde jsou obsažena všechna data o jednotlivých síťových zařízeních od počtu přečtených bytů, odeslaných bytů, vzniklé chyby až po rychlost tohoto síťového zařízení, čas pro

výpočet vzorků, jméno stroje a jméno daného síťového zařízení, na kterém byla tato data zachycena.

11.2.6 Další síťové statistiky (ipv4, icmpv4, udpv4, ipv6, icmpv6, udpv6)

Tyto entity nesou data o jednotlivých komunikačních protokolech ve verzi čtyři a šest. Nesou také informace svého vlastníka, jméno stroje, kterým byla data vygenerována. Nedílnou součástí je také čas ve kterém byla pořízena a čas předchozích dat.

11.2.7 Tcp statistiska

I tato entita nese data o komunikačním protokolu. Tento komunikační protokol se nazývá TCP. Kromě základních vlastních atribut nese také tato entita jméno stroje a časy.

12 Statistiky procesů

12.1 Hlavní statistika

I každý jednotlivý proces má svou statistiku, tato entita nese název procgen, tato statistika obsahuje data jako např. své id, id rodičovského procesu, jméno, čas běhu procesu a další. Tato entita je rozšířena dalšími několika entitami, které jsou:

- Proccpu zatížení procesoru procesem
- Procmem využití paměti procesem
- Procdsk zatížení disku procesem
- Procnet zatížení sítě procesem

Všechny tyto jednotlivé entity mají svého vlastníka a stroj který statistiku generoval. Vlastník znamená id procesu pid, čas předešlého a aktualního vzorku.

13 Konceptualní datový model

Návrhem konceptualního datového modelu nad jednotlivými entitami Atopu se zabývá tato kapitola. Je to nedílná součást tohoto projektu a navazuje na předešlou kapitolu Analýzu požadavků na databázi. Návrh je shrnut v následujících třech krocích:

- Popis vztahů jednotlivých entit.
- Výčet všech atributů pro jednotlivé entity, datové typy atributů.
- Výsledný konceptualní model.

13.1 Vztah entit

V případě, že entita obsahuje mnoho atributů je tento počet snížen. Pouze na ty atributy, které jsou pro zobrazení toho vztahu podstatné.

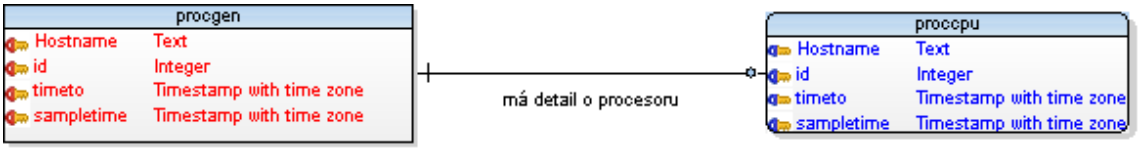
13.1.1 Obecné informace o procesech „procgen“- paměť využitá procesem „procmem“

Každý proces využívá určitou část z celkové volné paměti jak fyzické, swapu, stránky a to zobrazí právě jeden řádek v tabulce procmem.

Obrázek 4: vztah tabulek procgen a procmem

13.1.2 Obecné informace o procesech „procgen“ - využitý procesor procesem „proccpu“

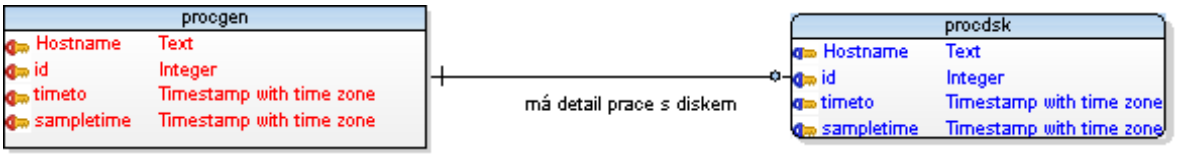
Všechny procesy vužívají procesorový čas, určité jádro nebo fyzický procesor. Tato informace je určena právě jedním řádkem v tabulce proccpu.



Obrázek 5: vztah tabulek procgen a proccpu

13.1.3 Obecné informace o procesech „procgen“ - využití disku procesem „procdsk“

Využití disku procesem je vyjádřeno vztahem tabulek procgen a procdsk, kdy řádek v tabulce procdsk zobrazí právě jednu informaci k procesu v čase.



Obrázek 6: vztah tabulek procgen a procdsk

13.1.4 Obecné informace o procesech „procgen“ - využití sítě procesem „procnet“

Když proces využívá jakékoliv síťové rozhraní je tento vztah zobrazen vztahem tabulek procgen a procnet.

Obrázek 7: vztah tabulek procgen a procnet

13.2 Shrnutí

Všechny tabulky procnet, procnet, proccpu, procnet musí být ve vztahu s jedním procesem, daný proces je určen id, časem aktuálního a předešlého vzorku, také jej určuje jméno stroje který statistiku vygeneroval.

13.3 Fyzický datový model

Předchozí kapitola detailně popisuje všechny elementy výsledného konceptuálního datového modelu. Tento výsledný datový model je nyní, možné převést do SQL kódu kompatibilního s databází PostgreSQL ve verzi 9.1. Tzn. ověření možných duplicit názvů a zvolení vhodných datových typů a syntaxe SQL jazyka.

13.4 Vytvoření databáze

Po vhodném sestrojení sql kódu databáze je možné provést tento kód nad databází vhodně zvoleným nástrojem. V případě, že databáze neshledá žádnou syntaktickou chybu provede vytvoření databáze.

14 Testování

K ověření správné funkčnosti upraveného Atopu dále jen Atop-klienta a přijímací strany (serveru) se dosáhlo testováním, které porovnávalo jednotlivé statistiky.Pro porovnání těchto statistik je zvolena aplikace SAR, která nad jednotlivými daty vytváří grafy.Tyto grafy jsou použity pro porovnání s grafy vytvořenými z dat generovánými Atop-klientem v jednoduché aplikaci LibreOffice Calc.

14.1 Sestavy určené k testování

Pro testování jsou zvoleny dvě sestavy:

- pro server tj. virtulání stroj s názvem vmmre01 a přidělenou konfigurací: CPU 2,5 Ghz,384 MB RAM,HDD 4GB.
- pro klienta je vybrán virtuální stroj vmmre02. Tento virtualní stroj má přidělenou stejnou hardwarovou konfiguraci jako server.

Na obou strojích je instalován operační systém CentOS Linux 5 (distribuce OS Linux binárně kompatibilní s RedHat Enterprise Linux 5).

14.2 Výsledky testování

Výsledky vybraných testování jsou uvedeny v příloze. U některých dat bylo nutné provést přepočítání, tak aby byly grafy generované Atopem a Sarem ve stejných jednotkách.

- Spotřeba procesorového času uživatelskými aplikacemi - zde je v jednotkách dvou grafů rozdíl proto jsou původní jednotky Atop-klienta přepočítány na procenta.
- Množství volné paměti - původní jednotky Atop-klienta jsou určeny jako počet stránek. Každá stránka má velikost 4096 Bytů, proto se pro dosažení sjednocení jednotek vynásobí velikost jedné stránky počtem stránek.
- U zbývajících třech měření celkového počtu I/O operací na disku sdb2, průměrná zátěž systému za poslední minutu a počtu přijatých paketů jednotky dvou grafů souhlasí, proto nejsou nutné další úpravy.

14.3 Zhodnocení dosažených výsledků

Malé rozdíly v těchto datech, mohou být způsobeny rozdílem času mezi dvěma vzorky, kdy vzorek aplikace SAR má délku jedné minuty. Rozdíl dvou vzorků Atop-klienta je 15 sekund. V grafech Atop-klienta lze vidět i skokový nárůst změny dat, tyto změny v aplikaci SAR nelze pozorovat. Test proběhl v době od 10:21 do 23:59. Na testovaných sestavách v tomto čase běžely, také další aplikace aby se napodobil standardní provoz. Strana klienta ani strana serveru nevykazovaly žádné přetěžování systému. To bylo nutné sledovat především na straně klienta která nesmí generovat značnou zátěž systému. Průměrná zátěž procesoru na testovaném stroji je do dvou procent, při potřebě 7 Mb celkové paměti.

15 Závěr

Cílem této bakalářské práce bylo rozšíření aplikace Atop o komunikaci server-klient a uložení statistik do databáze a dosažení zefektivnění práce s generovánými daty. Po analýze požadavků na rozšíření a seznámení se s aplikací Atop bylo přistoupeno k implementaci. Během vývoje rozšíření, došlo k několika hlavním změnám v aplikaci Atop především k zápisu na socket. Vývoj serverové části proběhl ve dvou fázích. První fáze měla umožnit zápis dat ze souboru genorovaného Atopem, jejich dekomprimací a zápis do databáze. Ve druhé fázi bylo doplněno čtení ze socketu a použita implementace zápisu do databáze. Rozšířením se splnily všechny požadavky vyplývající ze zadání. Je zde však mnoho možností, jak v budoucnu upravenou aplikaci rozvíjet. Jednou z těchto možností by mohlo být použití knihovny Lbasync, která by umožnila efektivnější způsob, jak binární data ukládat do databáze. Dalším rozšířením by mohla být kontrola spuštěných procesů na sledovaných strojích ze strany serveru, některé procesy by mohly mít nastavený limit na využité prostředky systému, po překročení tohoto limitu by byl proces “zabit”.

16 Reference

[1] Huffmanovo kódování. *h*ttp://planetmath.org [online]. 2013 [cit. 2013-04-20]. Dostupné z: http://planetmath.org/huffmansalgorithm

[2] Atop. *h*ttp://linux.die.net [online]. 2010 [cit. 2013-04-02]. Dostupné z: http://linux.die.net/man/1/atop

[3] Atop. *h*ttp://www.atoptool.nl [online]. 2010 [cit. 2013-04-02]. Dostupné z: http://www.atoptool.nl/index.php

[4] Top. *h*ttp://linux.die.net [online]. 2010 [cit. 2013-04-02]. Dostupné z: http://linux.die.net/man/1/top

[5] Htop. *h*ttp://linux.die.net [online]. 2010 [cit. 2013-04-02]. Dostupné z: http://linux.die.net/man/1/htop

[6] Powertop. *h*ttp://linux.die.net [online]. 2011 [cit. 2013-04-02]. Dostupné z: http://linux.die.net/man/1/Powertop

[7] Nmon. *h*ttp://nmon.sourceforge.net [online]. 20012 [cit. 2013-04-02]. Dostupné z: http://nmon.sourceforge.net/pmwiki.php

[8] Iotop. *h*ttp://linux.die.net [online]. 2010 [cit. 2013-04-02]. Dostupné z: http://linux.die.net/man/1/iotop

[9] SAR. *w*ww.root.cz [online]. 2012 [cit. 2013-04-02]. Dostupné z: http://www.root.cz/clanky/sar-sledovanie-vykonu-zbieranie-data-a-vytvaranie-statistik/

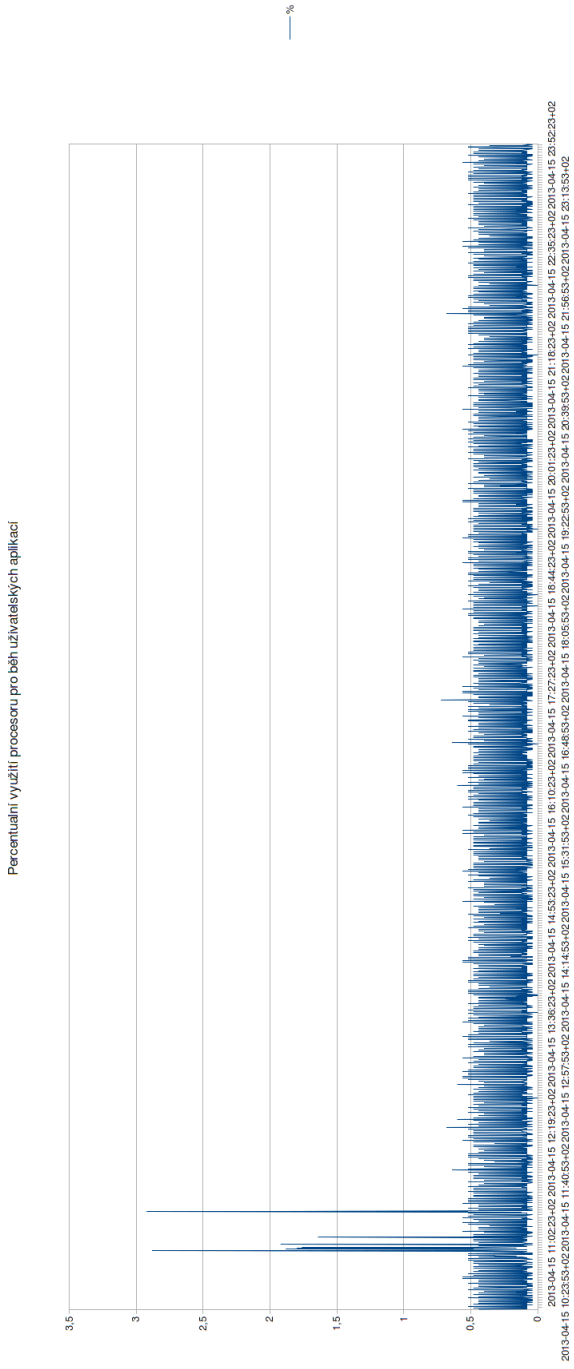
[10] Latencytop. *h*ttp://linux.die.net [online]. 2011 [cit. 2013-04-02]. Dostupné z: http://linux.die.net/man/8/latencytop

[11] Postgresql. *h*ttp://www.postgresql.org/ [online]. 2013 [cit. 2013-04-25]. Dostupné z: http://www.postgresql.org/docs/

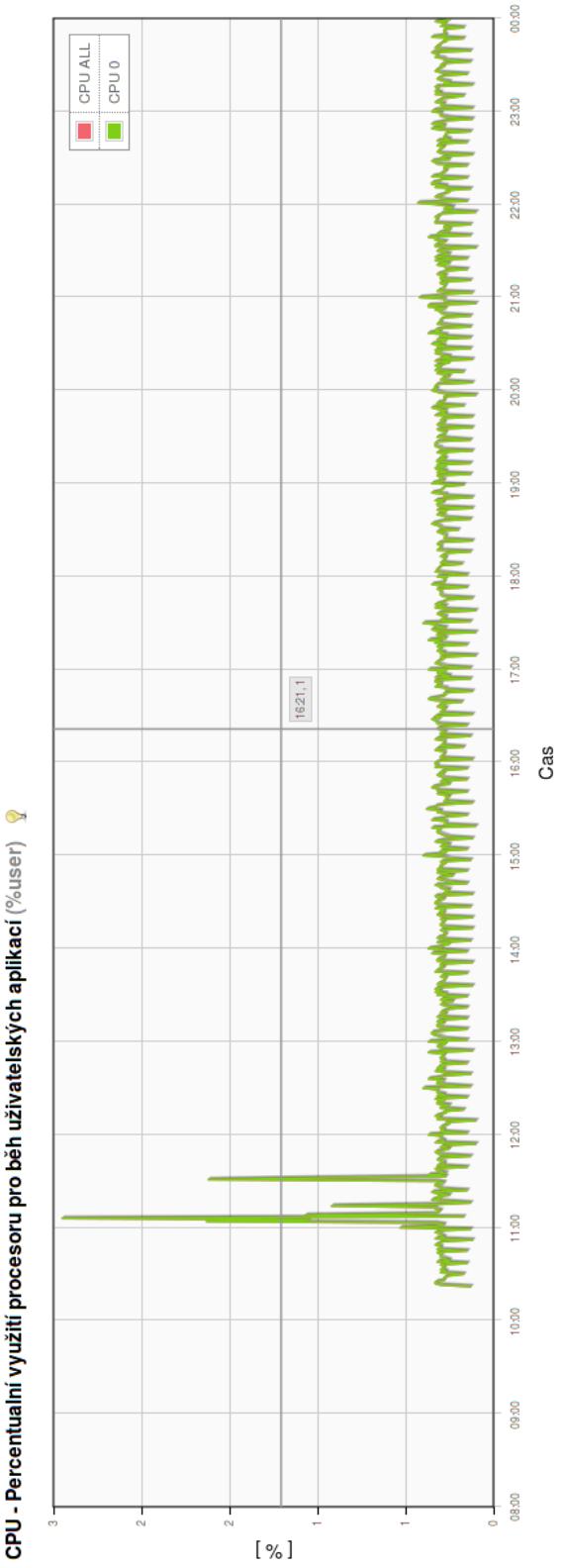
[12] Funkce select. *h*ttp://www.root.cz [online]. 2003 [cit. 2013-04-28]. Dostupné z: http://www.root.cz/clanky/sokety-a-c-funkce-select/

[13] Kompresní algoritmus LZ77. *h*ttp://voho.cz [online]. 2013 [cit. 2013-05-01]. Dostupné z: http://voho.cz/wiki/informatika/algoritmus/komprese/lz77/

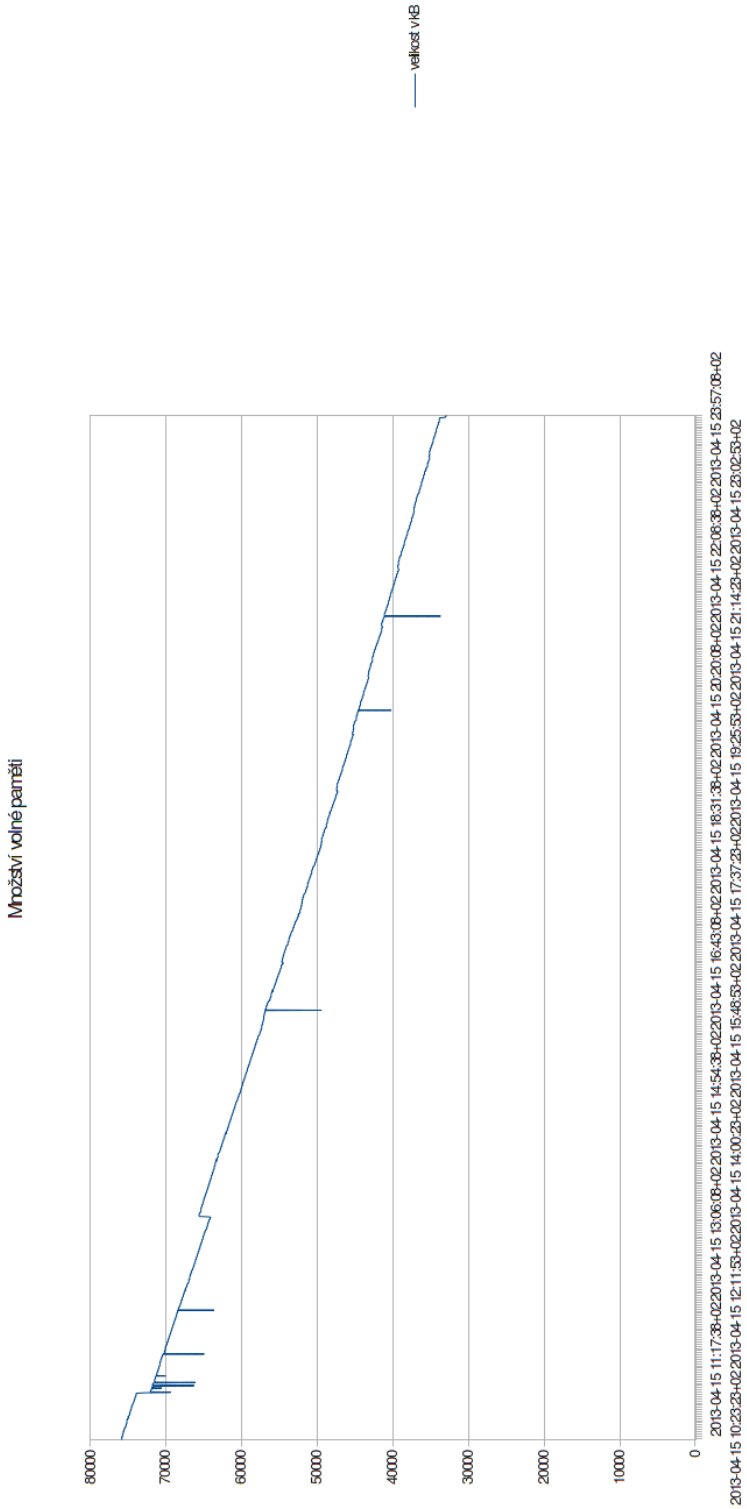
A Grafy



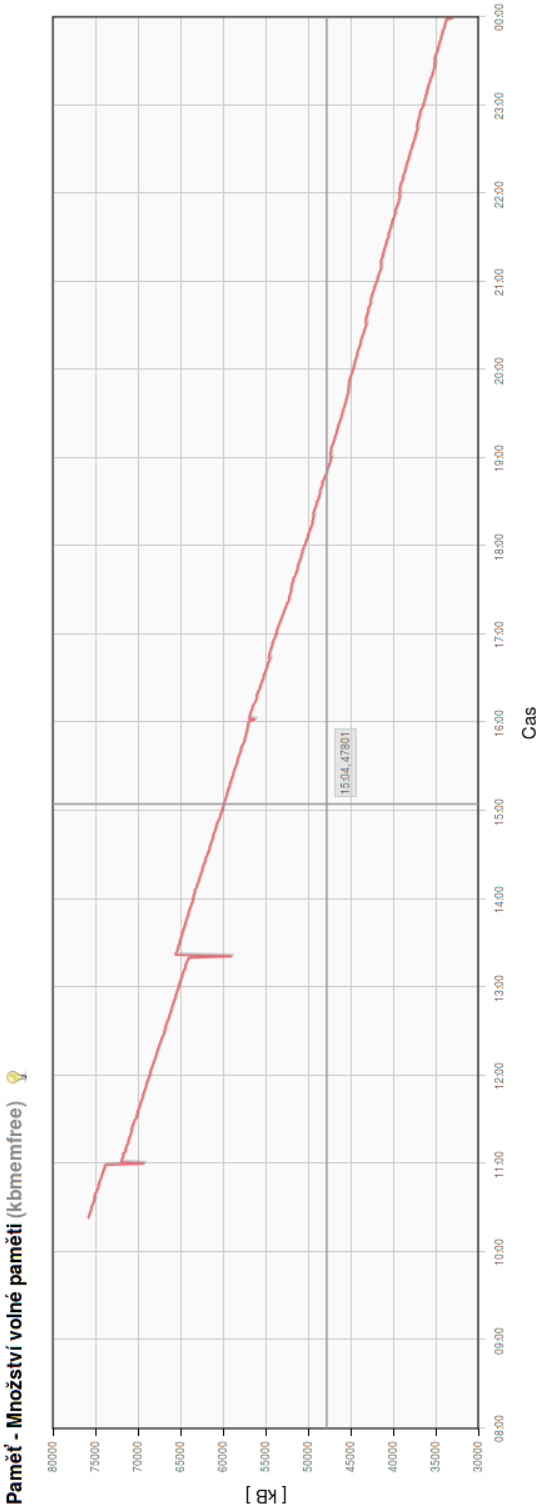
Obrázek 8: Procentualní využití procesoru pro běh uživatelských aplikací - Atop



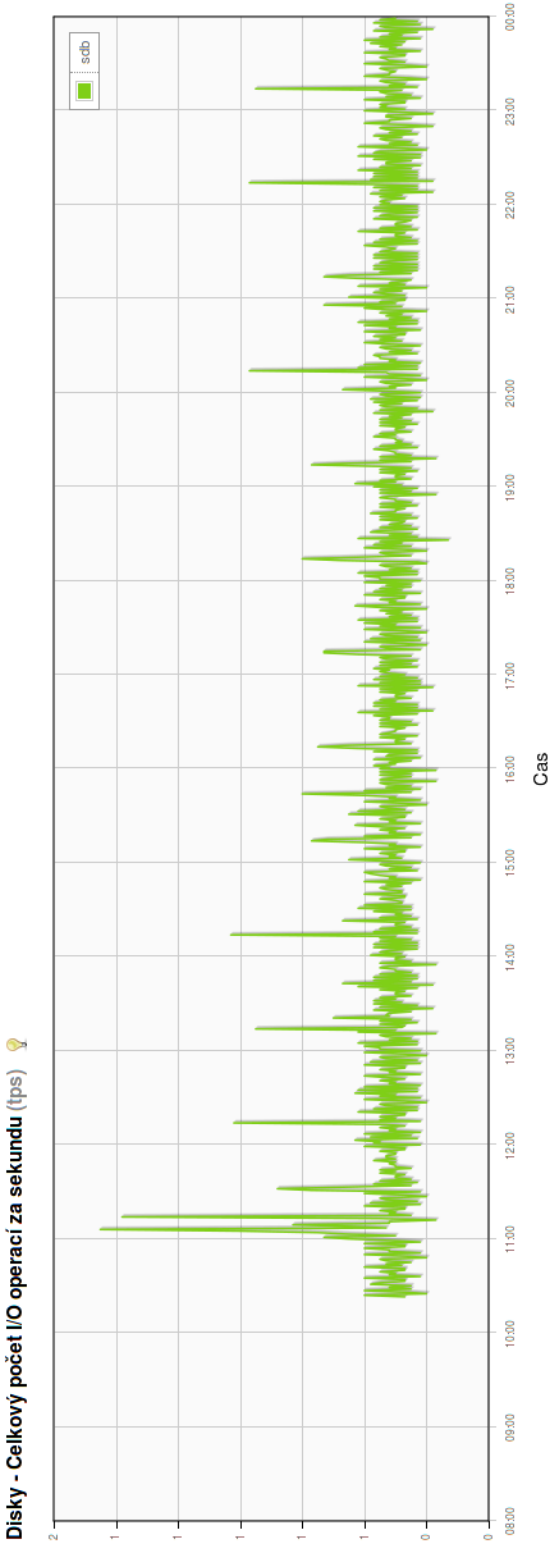
Obrázek 9: Procentualní využití procesoru pro běh uživatelských aplikací - SAR



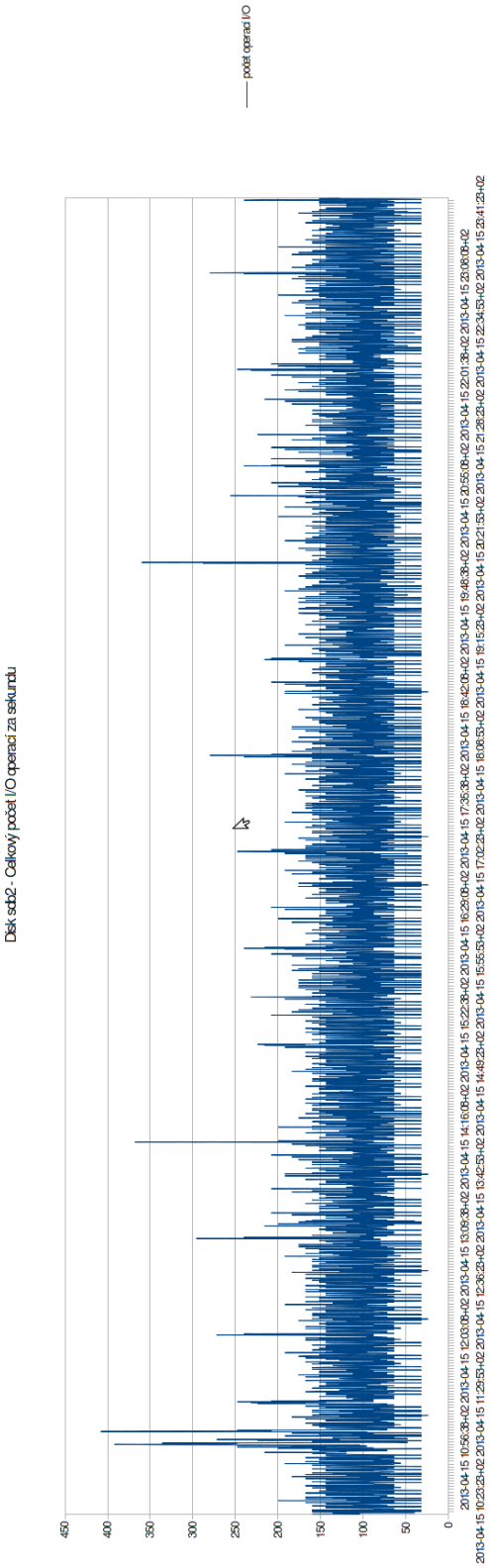
Obrázek 10: Množství volné paměti - Atop



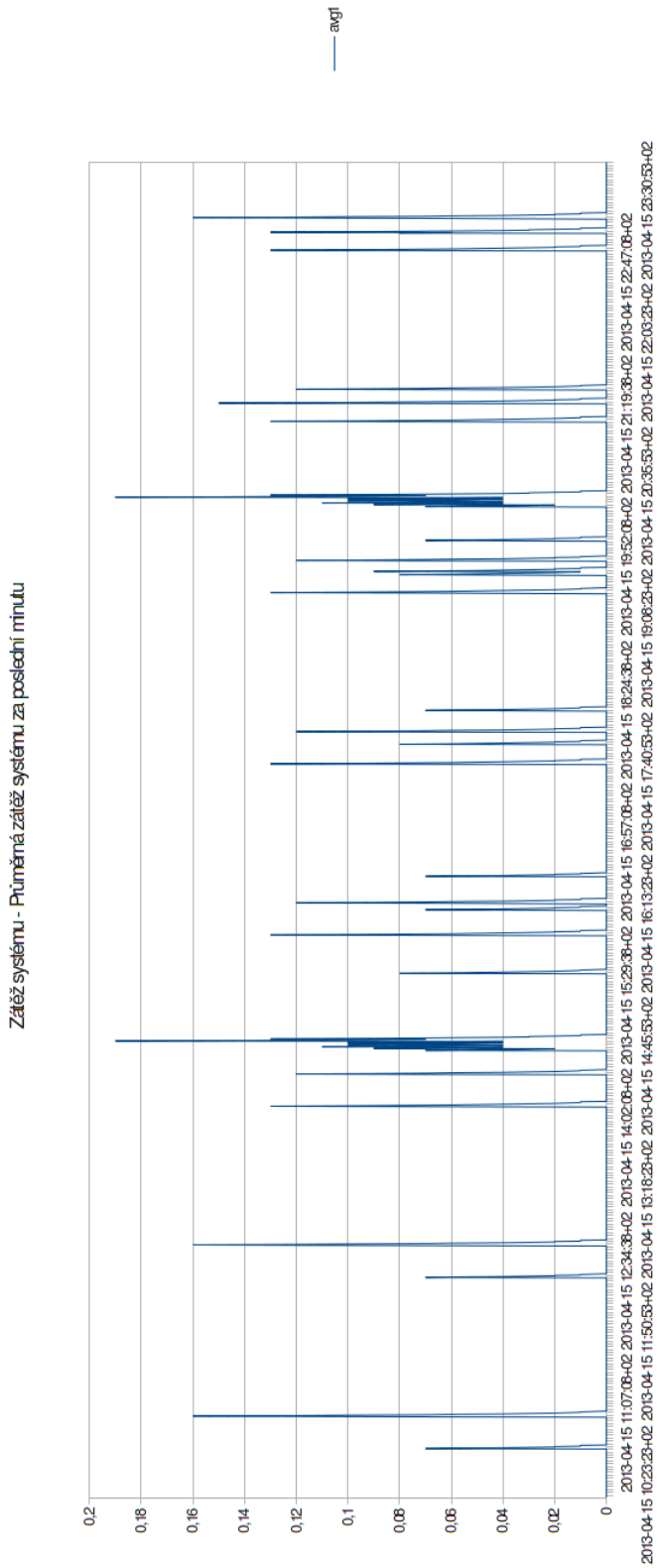
Obrázek 11: Množství volné paměti - SAR



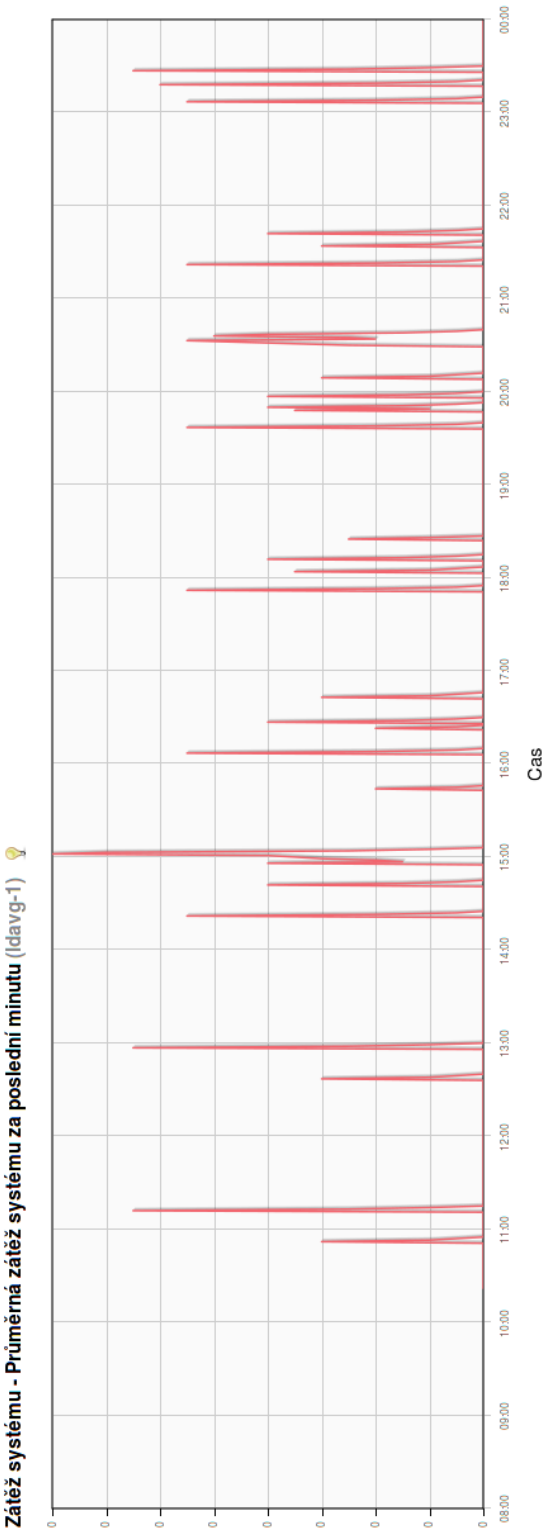
Obrázek 12: Disk - celkový počet I/O operací - Atop



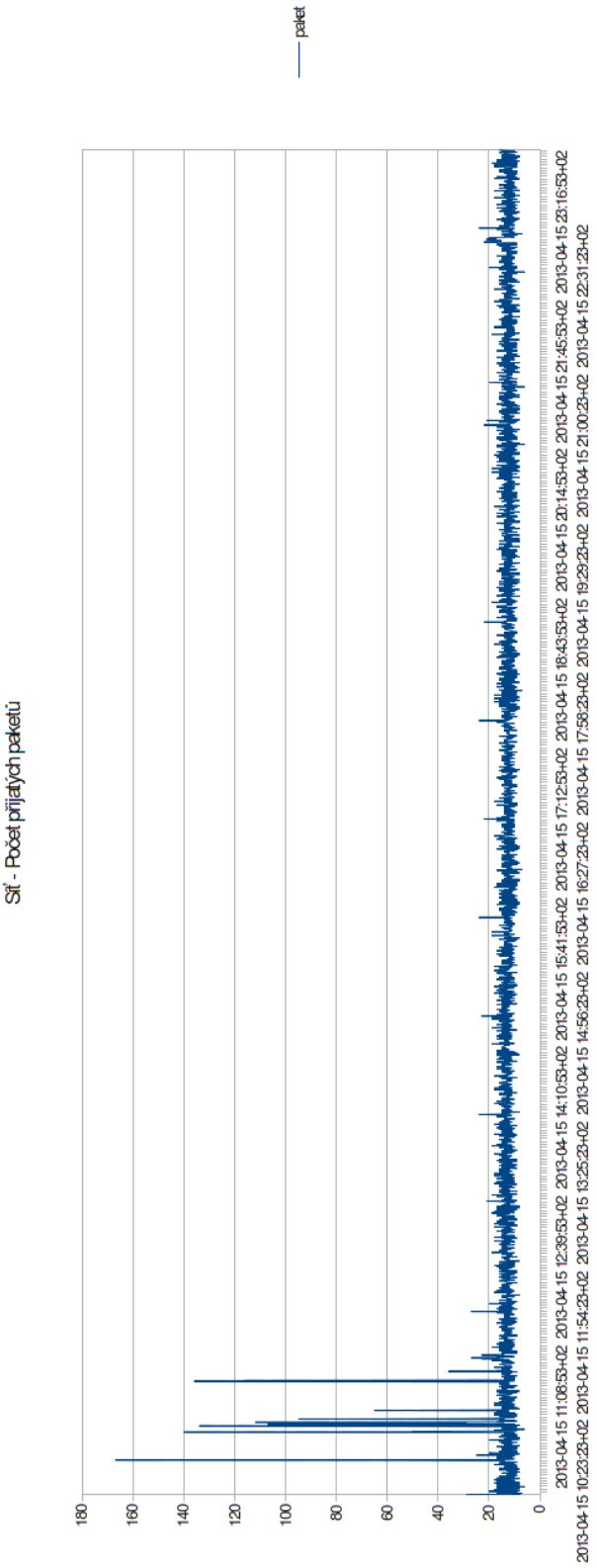
Obrázek 13: Disk - celkový počet I/O operací - SAR



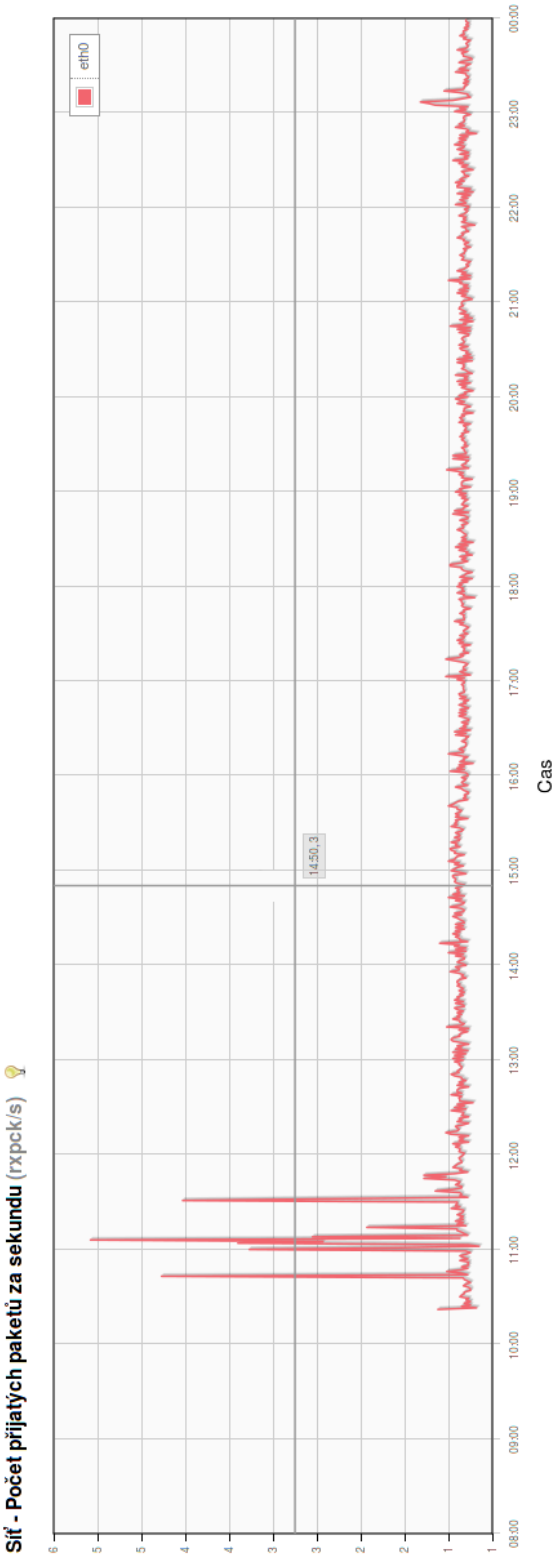
Obrázek 14: Zátěž systému - Atop



Obrázek 15: Zátěž systému - SAR



Obrázek 16: Počet přijatých paketů- Atop



Obrázek 17: Počet přijatých paketů- SAR

B SQL kód databáze

```
CREATE TABLE atop.cpustat
(
  hostname text NOT NULL,
  nrcpu bigint,
  devint bigint,
  csw bigint,
  nprocs bigint,
  lavg1 double precision,
  lavg5 double precision,
  lavg15 double precision,
  timeto timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT cpustat_pkey PRIMARY KEY (hostname, sampletime, timeto)
);


CREATE TABLE atop.dsk
(
  hostname text NOT NULL,
  dskname text NOT NULL,
  partype character(1),
  nread bigint,
  nrsect bigint,
  nwrite bigint,
  nwsect bigint,
  io_ms bigint,
  avque bigint,
  timeto timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT dsk_pkey PRIMARY KEY (hostname, dskname, timeto, sampletime)
);

CREATE TABLE atop.icmpv4stats
(
  hostname text NOT NULL,
  inmsgs bigint,
  inerrors bigint,
  indestunreachs bigint,
  intimeexcds bigint,
  inparmprobs bigint,
  insrcquenchs bigint,
  inredirects bigint,
  inechos bigint,
  inechoreps bigint,
  intimestamps bigint,
  intimestampreps bigint,
  inaddrmasks bigint,
  inaddrmaskreps bigint,
  outmsgs bigint,
```

```
  outerrors  bigint ,
  outdestunreachs bigint,
  outtimeexcds bigint,
  outparmprobs bigint,
  outsrcquenchs bigint,
  outredirects  bigint ,
  outechos bigint,
  outechoreps bigint,
  outtimestamps bigint,
  outtimestampreps bigint,
  outaddrmasks bigint,
  outaddrmaskreps bigint,
  timeto  timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT icmpv4stats_pkey PRIMARY KEY (hostname , timeto , sampletime )
) ;


CREATE TABLE atop.icmpv6stats
(
  hostname text NOT NULL,
  inmsgs bigint,
  inerrors  bigint ,
  indestunreachs bigint,
  inpkttoobigs  bigint ,
  intimeexcds bigint,
  inparmproblems bigint,
  inechos bigint,
  inechoreplies  bigint ,
  ingroupmembqueries bigint,
  ingroupmembresponses bigint,
  ingroupmembreductions bigint,
   inroutersolicits   bigint ,
  inrouteradvertisements bigint,
   inneighborsolicits  bigint ,
  inneighboradvertisements bigint,
   inredirects  bigint ,
  outmsgs bigint,
  outdestunreachs bigint,
  outpkttoobigs  bigint ,
  outtimeexcds bigint,
  outparmproblems bigint,
  outechoreplies bigint ,
   outroutersolicits   bigint ,
  outneighborsolicits  bigint ,
  outneighboradvertisements bigint,
  outredirects  bigint ,
  outgroupmembresponses bigint,
  outgroupmembreductions bigint,
  timeto  timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT icmpv6stats_pkey PRIMARY KEY (hostname , timeto , sampletime )
) ;
```

```
CREATE TABLE atop.intf
(
  hostname text NOT NULL,
  name text NOT NULL,
  rbyte  bigint ,
  rpack  bigint ,
  rerrs  bigint ,
  rdrop  bigint ,
  rfifo  bigint ,
  rframe  bigint ,
  rcompr bigint ,
  rmultic  bigint ,
  sbyte  bigint ,
  spack bigint ,
  serrs  bigint ,
  sdrop  bigint ,
  sfifo  bigint ,
  scollis  bigint ,
  scarrier  bigint ,
  scompr bigint,
  speed bigint,
  duplex "char",
  timeto  timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT intf_pkey PRIMARY KEY (hostname , name , timeto , sampletime )
) ;

CREATE TABLE atop.ipv4stats
(
  hostname text NOT NULL,
  forwarding  bigint ,
  defaultttl  bigint ,
  inreceives  bigint ,
  inhdrerrors  bigint ,
  inaddrerrors  bigint ,
  forwdatagrams bigint,
  inunknownprotos bigint,
  indiscards  bigint ,
  indelivers  bigint ,
  outrequests bigint ,
  outdiscards bigint ,
  outnoroutes bigint ,
  reasmtimeout bigint,
  reasmreqds bigint,
  reasmoks bigint,
  reasmfails  bigint ,
  fragoks  bigint ,
  fragfails  bigint ,
  fragcreates  bigint ,
  timeto  timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT ipv4stats_pkey PRIMARY KEY (hostname , timeto , sampletime )
) ;
```

```
CREATE TABLE atop.procgen
(
  hostname text NOT NULL,
  tgid  integer  NOT NULL,
  pid  integer  NOT NULL,
  ppid integer  NOT NULL,
  ruid  integer ,
  euid integer ,
  suid  integer ,
  fsuid  integer ,
  rgid  integer ,
  egid integer ,
  sgid  integer ,
  fsgid  integer ,
  nthr  integer ,
  procname text NOT NULL,
  state  text ,
  excode integer,
  btime timestamp with time zone NOT NULL,
  elaps timestamp with time zone,
  cmdline text ,
   nthrslpi   integer ,
  nthrsslpu  integer ,
  nthrrun  integer ,
  timeto  timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  id  integer  NOT NULL,
  CONSTRAINT procgen_pkey PRIMARY KEY (hostname , id , timeto , sampletime )
) ;

CREATE TABLE atop.ipv6stats
(
  hostname text NOT NULL,
  inreceives  bigint ,
  inhdrerrors  bigint ,
  intoobigerrors  bigint ,
  innoroutes bigint ,
  inaddrerrors  bigint ,
  inunknownprotos bigint,
  intruncatedpkts  bigint ,
  indiscards  bigint ,
   indelivers   bigint ,
  outforwdatagrams bigint,
  outrequests bigint ,
  outdiscards bigint ,
  outnoroutes bigint ,
  reasmtimeout bigint,
  reasmreqds bigint,
  reasmoks bigint,
  reasmfails  bigint ,
  fragoks  bigint ,
   fragfails   bigint ,
```

```
  fragcreates  bigint ,
  inmcastpkts bigint ,
  outmcastpkts bigint,
  timeto  timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT ipv6stats_pkey PRIMARY KEY (hostname , timeto , sampletime )
) ;

CREATE TABLE atop.memstat
(
  hostname text NOT NULL,
  physmem bigint,
  freemem bigint,
  buffermem bigint,
  slabmem bigint,
  cachemem bigint,
  cachedrt  bigint ,
  totswap  bigint ,
  freeswap bigint,
  pgscans bigint,
   allocstall   bigint ,
  swouts bigint,
  swins  bigint ,
  commitlim bigint,
  committ bigint,
  shmem bigint,
  shmrss bigint,
  shmswp bigint,
  slabreclaim  bigint ,
  timeto  timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT memstats_pkey PRIMARY KEY (hostname , timeto , sampletime )
) ;
CREATE TABLE atop.percpu
(
  hostname text NOT NULL,
  cpunum integer NOT NULL,
  stime  bigint ,
  utime  bigint ,
  ntime  bigint ,
  irqtime   bigint ,
  wtime bigint,
  itime   bigint ,
   softirqtime   bigint ,
  steal   bigint ,
  quest  bigint ,
  maxfreq bigint,
  cnt   bigint ,
   ticks   bigint ,
  timeto  timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT percpu_pkey PRIMARY KEY (hostname , cpunum , sampletime , timeto )
) ;
CREATE TABLE atop.proccpu
```

```
(
  hostname text NOT NULL,
  utime bigint,
  stime bigint,
  nice integer,
  prio integer,
  rtprio integer,
  policy integer,
  curcpu integer,
  sleepavg integer,
  timeto timestamp with time zone,
  sampletime timestamp with time zone,
  id integer,
  CONSTRAINT id_procgen FOREIGN KEY (hostname, id, timeto, sampletime)
      REFERENCES atop.procgen (hostname, id, timeto, sampletime) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
);

CREATE TABLE atop.procdsk
(
  hostname text NOT NULL,
  rsz bigint,
  wio bigint,
  wsz bigint,
  cwsz bigint,
  timeto timestamp with time zone,
  sampletime timestamp with time zone,
  id integer,
  CONSTRAINT id_procgen FOREIGN KEY (hostname, id, timeto, sampletime)
      REFERENCES atop.procgen (hostname, id, timeto, sampletime) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
);



CREATE TABLE atop.procmem
(
  hostname text NOT NULL,
  minflt bigint,
  majflt bigint,
  shtext bigint,
  vmem bigint,
  rmem bigint,
  vgrow bigint,
  rgrow bigint,
  vdata bigint,
  vstack bigint,
  vlibs bigint,
  vswap bigint,
  timeto timestamp with time zone,
  sampletime timestamp with time zone,
  id integer,
  CONSTRAINT id_procgen FOREIGN KEY (hostname, id, timeto, sampletime)
      REFERENCES atop.procgen (hostname, id, timeto, sampletime) MATCH SIMPLE
```

```
        ON UPDATE NO ACTION ON DELETE NO ACTION
);

CREATE TABLE atop.procnet
(
  hostname text NOT NULL,
  tcpsnd bigint,
  tcpssz bigint,
  tcprcv bigint,
  tcprsz bigint,
  udpsnd bigint,
  udpssz bigint,
  avail1 bigint,
  avail2 bigint,
  timeto timestamp with time zone,
  sampletime timestamp with time zone,
  id integer,
  CONSTRAINT id_procgen FOREIGN KEY (hostname, id, timeto, sampletime)
      REFERENCES atop.procgen (hostname, id, timeto, sampletime) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
);

CREATE TABLE atop.tcpstats
(
  hostname text NOT NULL,
  rtoalgorithm bigint,
  rtomin bigint,
  rtomax bigint,
  maxconn bigint,
  activeopens bigint,
  passiveopens bigint,
  attemptfails bigint,
  estabresets bigint,
  currestab bigint,
  insegs bigint,
  outsegs bigint,
  retranssegs bigint,
  inerrs bigint,
  outrsts bigint,
  timeto timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT tcpstats_pkey PRIMARY KEY (hostname, timeto, sampletime)
);

CREATE TABLE atop.udpv6stats
(
  hostname text NOT NULL,
  indatagrams bigint,
  noports bigint,
  inerrors bigint,
  outdatagrams bigint,
  timeto timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT udpv6stats_pkey PRIMARY KEY (hostname, timeto, sampletime)
```

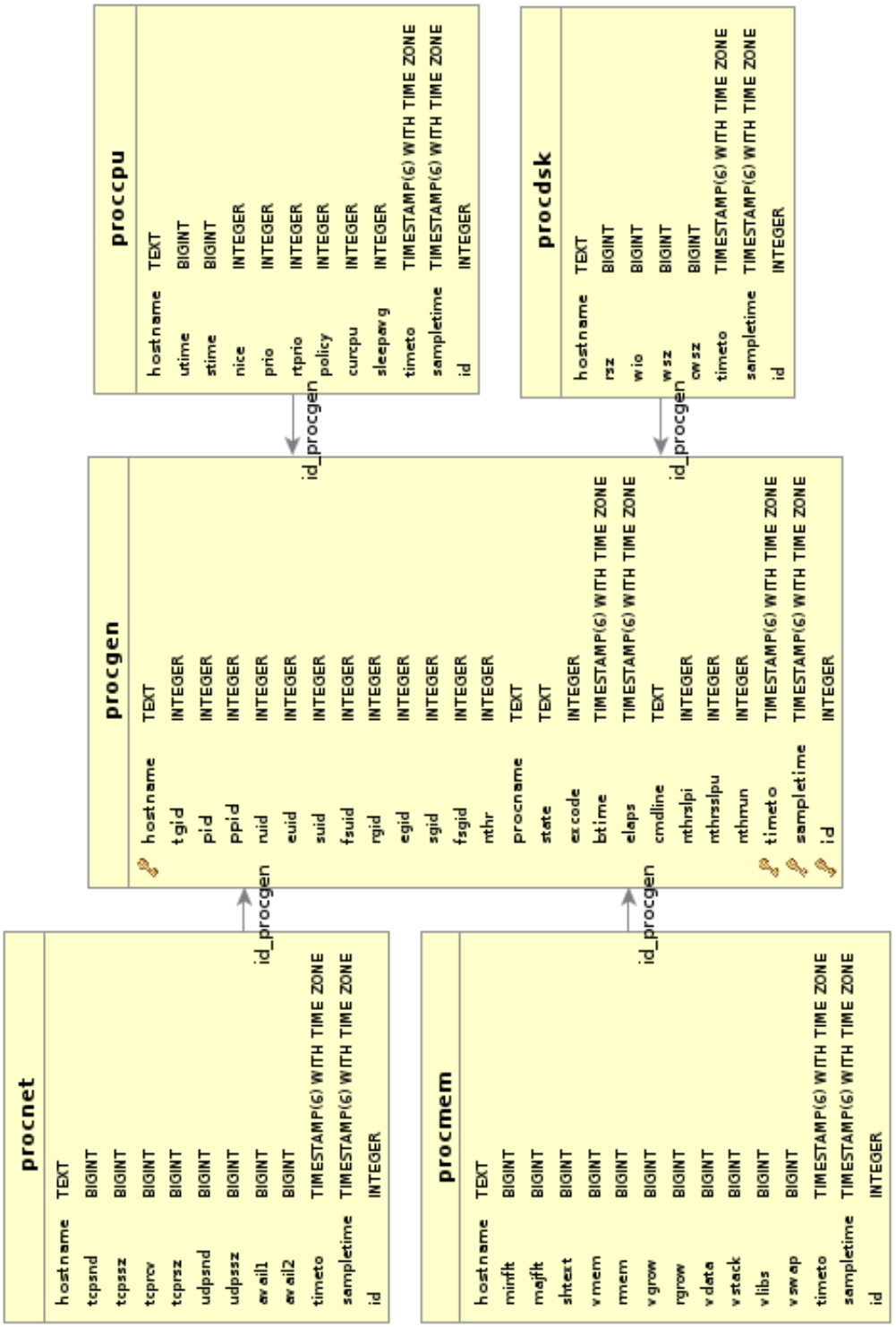
```
);

CREATE TABLE atop.udpv4stats
(
  hostname text NOT NULL,
  indatagrams bigint,
  noports  bigint ,
  inerrors   bigint ,
  outdatagrams bigint,
  timeto  timestamp with time zone NOT NULL,
  sampletime timestamp with time zone NOT NULL,
  CONSTRAINT udpv4stats_pkey PRIMARY KEY (hostname , timeto , sampletime )
);
```

Výpis 1: SQL kód databáze

C Datový model

Obrázek 18: Datový model



Obrázek 19: Datový model

Obrázek 20: Datový model

D Popis nejvýznamnějších struktur

Paměť	
physmem	number of physical pages
freemem	number of free pages
buffermem	number of buffer pages
slabmem	number of slab pages
cachemem	number of cache pages
cachedrt	number of cache pages (dirty)
totswap	number of pages in swap
freeswap	number of free swap pages
pgscans	number of page scans
allocstall	try to free pages forced
swouts	number of pages swapped out
swins	number of pages swapped in

Tabulka 1: Struktury paměti

Procesor	
maxfreq	frequency in MHz
cnt	number of clock ticks times state
ticks	number of total clock ticks
stime	system time in clock ticks
utime	user time in clock ticks
ntime	nice time in clock ticks
itime	idle time in clock ticks
wtime	iowait time in clock ticks
Itime	irq time in clock ticks
Stime	softirq time in clock ticks
steal	steal time in clock ticks
guest	guest time in clock ticks

Tabulka 2: Struktury procesoru

System	
nrcpu	number of cpu's
devint	number of device interrupts
csw	number of context switches
nprocs	number of processes started
avg1	load average last minute
avg5	load average last 5 minutes
lavg15	load average last 15 minutes

Tabulka 3: Struktury systemu

Disky	
name	empty string for last
nread	number of read transfers
nrsect	number of sectors read
nwrite	number of write transfers
nwsect	number of sectors written
io_ms	number of millisecs spent for I/O
avque	average queue length

Tabulka 4: Struktury disky

Sítě	
name	empty string for last
rbyte	number of read bytes
rpack	number of read packets
rerrs	receive errors
rdrop	receive drops
rfifo	receive fifo
rframe	receive framing errors
rcompr	receive compressed
rmultic	receive multicast
sbyte	number of written bytes
spack	number of written packets
serrs	transmit errors
sdrop	transmit drops
sfifo	transmit fifo
scollis	collisions
scarrier	transmit carrier
scompr	transmit compressed
speed	interface speed in megabits/second
duplex	full duplex (boolean)

Tabulka 5: Struktury sítě

Zatížení paměti procesem	
minflt	number of page-reclaims
majflt	number of page-faults
shtext	text memory (Kb)
vmem	virtual memory (Kb)
rmem	resident memory (Kb)
vgrow	virtual growth (Kb)
rgrow	resident growth (Kb)

Tabulka 6: Struktury procesů - Zatížení paměti procesem

Zatížení procesoru procesem	
utime	time user text (ticks)
stime	time system text (ticks)
nice	nice value
prio	priority
rtprio	realtime priority
policy	scheduling policy
curcpu	current processor
sleepavg	sleep average percentage

Tabulka 7: Struktury procesů - Zatížení procesoru procesem

Procesy obecné informace	
pid	process identification
ppid	parent process identification
ruid	real user identification
euid	eff. user identification
suid	saved user identification
fsuid	fs user identification
rgid	real group identification
egid	eff. group identification
sgid	saved group identification
fsgid	fs group identification
nthr	number of threads in tgroup
name	process name string
state	process state ('E' = exited)
excode	process exit status
btime	process start time (epoch)
elaps	process elaps time (hertz)
cmdline	command-line string
nthrslpi	threads in state 'S'
nthrslpu	threads in state 'D'
nthrrun	threads in state 'R'

Tabulka 8: Struktury procesů - obecné informace

Zatížení disku procesem	
rio	number of read requests
rsz	cumulative sectors read
wio	number of write requests
wsz	cumulative sectors written
cwsz	cumulative written sectors

Tabulka 9: Struktury procesů - zatížení disku procesem

Zatížení sítě procesem	
tcpsnd	number of TCP-packets sent
tcpssz	cumulative size packets sent
tcprcv	number of TCP-packets recved
tcprsz	cumulative size packets rcvd
udpsnd	number of UDP-packets sent
udpssz	cumulative size packets sent
udprcv	number of UDP-packets recved
udprsz	cumulative size packets sent
rawsnd	number of raw packets sent
rawrcv	number of raw packets recved

Tabulka 10: Struktury procesů - zatížení sítě procesem